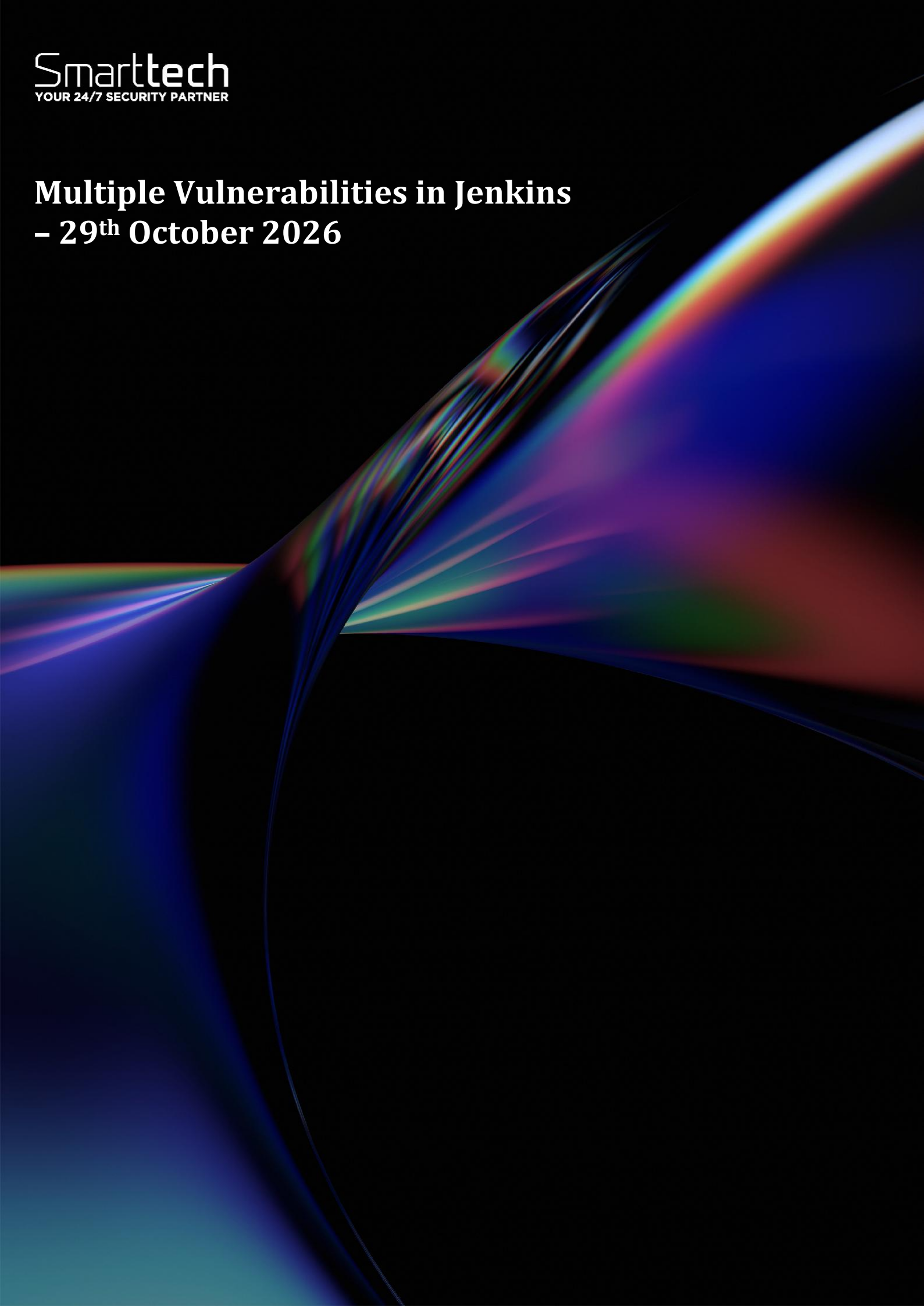
Smarttech
YOUR 24/7 SECURITY PARTNER

# Multiple Vulnerabilities in Jenkins – 29th October 2026



| | |
|---|---|
| **Document ID** | SMA-Threat Report |
| **Document status** | ISSUED |
| **Issue Number** | 189 |
| **Authors** | Tudor Anton < tudor.anton@smarttech247.com > |
| **Verified by** | Alin Curcan < alin.curcan@smarttech247.com > |
| **Last modified** | 2026-10-29 |
| **Issue Date** | 2026-10-28 |

This document and any information therein are confidential property of Smarttech247 and without infringement neither the whole nor any extract may be disclosed, loaned, copied or used for manufacturing, provision of services or other purposes whatsoever without prior written consent of Smarttech247, and no liability is accepted for loss or damage from any cause whatsoever from the use of the document. Smarttech247 retain the right to alter the document at any time unless a written statement to the contrary has been appended.

**Threat Reports** are reports created by Smarttech247 based on high and critical severity vulnerabilities that may have a high potential to be exploited in the wild i.e. vulnerabilities that are present in most used products by companies and do not have an auto-update option or they are usually not automatically updated in case that could lead to some service disruption. This report is usually created as soon as the vulnerability is released, therefore we strongly recommend that the information is reviewed, tests are performed and patches are applied before the first proof-of-concept is released. Even though certain vulnerabilities may not have an active exploit in the wild at the time that we report on them, we take into consideration the wider risk and the impact it could have on systems, should an exploit like that be available after a while. Our duty is to report them on time and we recommend enterprises that, in order to keep critical business systems protected, they should consider, on average, ten working days to check whether or not the new vulnerability affects them, and if so, to implement actions in order to remove the risk.

## Overview:

Multiple vulnerabilities have been identified in Jenkins. Successful exploitation of these vulnerabilities could lead to arbitrary code execution, information disclosure, server-side request forgery (SSRF), cross-site scripting (XSS), and file-system manipulation.

### Risk

Government:

- Large and medium government entities: **High**
- Small government entities: **High**

Businesses:

- Large and medium business entities: **High**
- Small business entities: **High**

## Technical summary

More details related to these vulnerabilities are as follows:

| ***CVE ID*** | ***Description*** |
|---|---|
| RCE vulnerability in XStream deserialization due to improper `resolves-to` attribute handling<br>**CVE-2026-117612**<br>CVSS Base Score: 8.8 | Jenkins uses XStream for serialization and deserialization of configuration and build data. To protect from common deserialization vulnerabilities, Jenkins uses a custom deserialization filter that only allows deserialization of types defined in Jenkins core or plugins, or explicitly allowed types (JEP-200). These vulnerabilities generally rely on behavior defined in #readResolve methods that are executed during deserialization.<br>The custom XStream converter used by Jenkins 2.587 and earlier, LTS 2.580.1 and earlier does not apply the deserialization filter to classes referenced by the resolves-to attribute. This allows attackers to instantiate arbitrary types during deserialization, potentially leading to remote code execution. To do this, attackers need to have Overall/Read permission, and at least one of the following:<br>• a user account (i.e., cannot be anonymous)<br>• any set of permissions allowing them to POST config.xml (e.g., Item/Configure, View/Configure, Agent/Configure). |

| | |
|---|---|
| | Jenkins 2.588, LTS 2.580.2 applies the deserialization filter to classes referenced by the resolves-to attribute. Additionally, Jenkins 2.588, LTS 2.580.2 newly uses the XStream type permissions system (in addition to its own custom converters) to enforce allowed types during deserialization. |
| Collection element type check bypass vulnerability<br>**CVE-2026-117613**<br>CVSS Base Score: 8 | Jenkins uses XStream for serialization and deserialization of configuration and build data. The ability to provide XML documents to XStream is exposed to users who hold the respective */Configure permission, e.g., the item POST config.xml API is available with Item/Configure permission.<br>XStream by itself does not restrict the types of collection elements after deserialization. While some converters in Jenkins restrict collection element types (see SECURITY-3707 in the 2026-06-10 security advisory), not all converters restrict element types.<br>In Jenkins 2.587 and earlier, LTS 2.580.1 and earlier this allows attackers with Item/Configure permission to submit a crafted job config.xml file that causes a stored cross-site scripting (XSS) vulnerability. |
| Agent launch command approval bypass vulnerability<br>**CVE-2026-117614**<br>CVSS Base Score: **8.8** | Since Jenkins 2.86, LTS 2.89.1 launching agents by running a command on the controller is implemented in the Command Agent Launcher Plugin rather than in Jenkins.<br>For backward compatibility with older agent configurations, Jenkins 2.587 and earlier, LTS 2.580.1 and earlier converts the launch commands they contain into launchers provided by this plugin.<br>This conversion does not check whether the launch command has been approved by an administrator, allowing attackers with Agent/Configure permission to execute arbitrary code in the context of the Jenkins controller JVM.<br>Jenkins 2.588, LTS 2.580.2 no longer converts the launch commands in older agent configurations into launchers provided by the Command Agent Launcher Plugin. |
| Link following vulnerability allows arbitrary file read<br>**CVE-2026-117615**<br>CVSS Base Score: 8.8 | When renaming an item, Jenkins moves its directory on the controller file system, or copies it if it cannot be moved.<br>Jenkins 2.587 and earlier, LTS 2.580.1 and earlier follows symbolic links when copying the directory of an item being renamed.<br>This allows attackers with Item/Create and Item/Configure permission to read arbitrary files on the Jenkins controller file system, which can lead to remote code execution (see Reading Files). |

| | |
|---|---|
| | Jenkins 2.588, LTS 2.580.2 no longer copies the directory of an item being renamed if it contains symbolic links. |
| Path traversal vulnerability in agent names<br>**CVE-2026-117616**<br>CVSS Base Score: **8.8** | Jenkins 2.587 and earlier, LTS 2.580.1 and earlier provides an HTTP endpoint that loads agents by name without validating the name, so that any config.xml file on the controller file system can be loaded, as an agent carrying the name declared in that file.<br>This allows attackers with Overall/Read permission to load any config.xml file on the controller file system as an agent configuration, replacing the configuration of an existing agent or defining a new agent. The name declared in that file also determines where the agent's configuration is written. When this agent is saved, Jenkins overwrites the config.xml file at that location, potentially resulting in arbitrary code execution.<br>Jenkins 2.588, LTS 2.580.2 no longer allows loading agents by name through HTTP requests.<br>Additionally, when agents are loaded by name, it now validates the name and rejects configurations whose declared name differs from the requested name. |
| Race condition in artifact extraction allows arbitrary file creation<br>**CVE-2026-117617**<br>CVSS Base Score: **8.8** | To prevent writing files outside the target directory, Jenkins refuses to write files through symbolic links when extracting archived artifacts on the controller.<br>Jenkins 2.587 and earlier, LTS 2.580.1 and earlier checks for symbolic links in the path of an extracted file only before creating its parent directories, not immediately before writing it, and allows symbolic links in artifacts to replace existing files or empty directories.<br>This allows attackers with Item/Configure permission to write files to arbitrary locations on the Jenkins controller file system by archiving artifacts concurrently, e.g., from parallel branches of a Pipeline, which can lead to remote code execution (see Writing Files). |
| XSS vulnerability in workspace browser<br>**CVE-2026-117621**<br>CVSS Base Score: **8** | When the workspace of a job is located on an agent, Jenkins retrieves the list of files and directories shown in the workspace browser from that agent.<br>In Jenkins 2.587 and earlier, LTS 2.580.1 and earlier, the agent computes the URL of the link shown for each file and directory.<br>This results in a cross-site scripting (XSS) vulnerability exploitable by attackers in control of agent processes.<br>Jenkins 2.588, LTS 2.580.2 computes on the controller the URL of the link shown for each file and directory in the workspace browser. |
| XSS vulnerability in handling of agent errors<br>**CVE-2026-117622**<br>CVSS Base Score: **8** | To protect against deserialization vulnerabilities, Jenkins restricts the types of objects that agents can send to the controller (JEP-200).<br>Jenkins 2.587 and earlier, LTS 2.580.1 and earlier allows agents to send exceptions implementing the |

| | |
|---|---|
| | Stapler HttpResponse interface, and renders them as HTTP responses.<br>This results in a cross-site scripting (XSS) vulnerability exploitable by attackers in control of agent processes.<br>Jenkins 2.588, LTS 2.580.2 does not allow agents to send exceptions implementing the Stapler HttpResponse interface. Additionally, Jenkins 2.588, LTS 2.580.2 sets a restrictive Content-Security-Policy header on HTTP responses for form fill failures. |
| Stored XSS vulnerability in breadcrumb bar overflow menu<br>**CVE-2026-117623**<br>CVSS Base Score: **8** | When the breadcrumb bar does not fit in the browser window, Jenkins moves breadcrumbs into an overflow menu, and shows the full text of long breadcrumbs, such as job, build, or user names, in a tooltip.<br>Jenkins 2.587 and earlier, LTS 2.580.1 and earlier does not escape the text shown in these tooltips.<br>This results in a stored cross-site scripting (XSS) vulnerability exploitable by attackers with Overall/Read permission.<br>Jenkins 2.588, LTS 2.580.2 escapes the text shown in the tooltips of breadcrumb bar overflow menu entries |
| Race condition in setup wizard allows creating an administrator account<br>**CVE-2026-117618**<br>CVSS Base Score: **8.1** | To prevent administrators from being locked out, the "Jenkins' own user database" security realm provides an HTTP endpoint that allows unauthenticated users to create a user account while no user accounts exist.<br>In Jenkins 2.587 and earlier, LTS 2.580.1 and earlier, this HTTP endpoint does not check whether the setup wizard has been completed, resulting in a race condition that allows unauthenticated attackers to create an administrator account while the initial account is being replaced. |
| Insufficient session expiration vulnerability<br>**CVE-2026-117619**<br>CVSS Base Score: **8.1** | Jenkins 2.587 and earlier, LTS 2.580.1 and earlier allows requests authenticated using HTTP Basic authentication, including with an API token, to obtain a "remember me" cookie, allowing attackers who have obtained a user's API token to retain access to Jenkins using a "remember me" cookie after the API token is revoked or expires. |
| Old data Monitor injection lets users take over administrator accounts<br>**CVE-2026-117620**<br>CVSS Base Score: **8** | Jenkins 2.587 and earlier, LTS 2.580.1 and earlier allows attackers with Overall/Read permission to trigger warnings in the "Manage Old Data" feature that can be acted on by administrators to inject attacker-controlled data, overriding existing user account information, including login passwords. |
| Root element restriction bypass in Xstream deseriliazation<br>**CVE-2026-117624**<br>CVSS Base Score: **7.7** | Jenkins 2.587 and earlier, LTS 2.580.1 and earlier does not consistently enforce the restriction on deserializing objects of types marked as storing their configuration in independent top-level configuration files as nested values, allowing attackers to bypass the root element check and potentially inject nested objects. |

| | |
|---|---|
| Missing permission check allows obtaining HTTP proxy configuration<br>**CVE-2026-117625**<br>CVSS Base Score: **4.3** | A missing permission check in Jenkins 2.587 and earlier, LTS 2.580.1 and earlier allows attackers with Overall/Read permission to obtain the server, port, user name, and no-proxy host list of the configured HTTP proxy. |
| Path traversal vulnerability in file browser<br>**CVE-2026-117626**<br>CVSS Base Score: **4.3** | Jenkins 2.587 and earlier, LTS 2.580.1 and earlier improperly identifies file paths attempting path traversal in the file browser for workspaces and $JENKINS_HOME/userContent/, allowing attackers with Overall/Read permission to obtain the names, sizes, and last modification times of files in the $JENKINS_HOME, and attackers with Item/Workspace permission to obtain the names of jobs that have a workspace on the same agent. |

## Affected Products:

- Jenkins weekly up to and including 2.587
- Jenkins LTS up to and including 2.580.1

- Fixed Versions:

- Jenkins weekly should be updated to version 2.588
- Jenkins LTS should be updated to version 2.580.2

## Recommendations

**Smarttech247** team recommend the following actions to be taken:

- Apply appropriate updates provided by Jenkins to vulnerable systems immediately after appropriate testing. (M1051: Update Software)
    - Safeguard 7.1 : Establish and Maintain a Vulnerability Management Process: Establish and maintain a documented vulnerability management process for enterprise assets. Review and update documentation annually, or when significant enterprise changes occur that could impact this Safeguard.
    - Safeguard 7.2: Establish and Maintain a Remediation Process: Establish and maintain a risk-based remediation strategy documented in a remediation process, with monthly, or more frequent, reviews.
    - Safeguard 7.4: Perform Automated Application Patch Management: Perform application updates on enterprise assets through automated patch management on a monthly, or more frequent, basis.
    - Safeguard 7.5 : Perform Automated Vulnerability Scans of Internal Enterprise Assets: Perform automated vulnerability scans of internal enterprise assets on a quarterly, or more frequent, basis. Conduct both authenticated and unauthenticated scans, using a SCAP-compliant vulnerability scanning tool.
    - Safeguard 7.7: Remediate Detected Vulnerabilities: Remediate detected vulnerabilities in software through processes and tooling on a monthly, or more frequent, basis, based on the remediation process.
    - Safeguard 12.1: Ensure Network Infrastructure is Up-to-Date: Ensure network infrastructure is kept up-to-date. Example implementations include running the latest stable release of software and/or using currently supported network-as-a-

service (NaaS) offerings. Review software versions monthly, or more frequently, to verify software support.
    - Safeguard 18.1: Establish and Maintain a Penetration Testing Program: Establish and maintain a penetration testing program appropriate to the size, complexity, and maturity of the enterprise. Penetration testing program characteristics include scope, such as network, web application, Application Programming Interface (API), hosted services, and physical premise controls; frequency; limitations, such as acceptable hours, and excluded attack types; point of contact information; remediation, such as how findings will be routed internally; and retrospective requirements.
    - Safeguard 18.2: Perform Periodic External Penetration Tests: Perform periodic external penetration tests based on program requirements, no less than annually. External penetration testing must include enterprise and environmental reconnaissance to detect exploitable information. Penetration testing requires specialized skills and experience and must be conducted through a qualified party. The testing may be clear box or opaque box.
    - Safeguard 18.3: Remediate Penetration Test Findings: Remediate penetration test findings based on the enterprise's policy for remediation scope and prioritization.
- Vulnerability scanning is used to find potentially exploitable software vulnerabilities to remediate them. (M1016: Vulnerability Scanning)
    - Safeguard 16.13: Conduct Application Penetration Testing: Conduct application penetration testing. For critical applications, authenticated penetration testing is better suited to finding business logic vulnerabilities than code scanning and automated security testing. Penetration testing relies on the skill of the tester to manually manipulate an application as an authenticated and unauthenticated user.
- Apply the Principle of Least Privilege to all systems and services. Run all software as a non-privileged user (one without administrative privileges) to diminish the effects of a successful attack. (M1026: Privileged Account Management)
    - Safeguard 4.7: Manage Default Accounts on Enterprise Assets and Software: Manage default accounts on enterprise assets and software, such as root, administrator, and other pre-configured vendor accounts. Example implementations can include: disabling default accounts or making them unusable.
    - Safeguard 5.4: Restrict Administrator Privileges to Dedicated Administrator Accounts: Restrict administrator privileges to dedicated administrator accounts on enterprise assets. Conduct general computing activities, such as internet browsing, email, and productivity suite use, from the user's primary, non-privileged account.
    - Safeguard 5.5: Establish and Maintain an Inventory of Service Accounts: Establish and maintain an inventory of service accounts. The inventory, at a minimum, must contain department owner, review date, and purpose. Perform service account reviews to validate that all active accounts are authorized, on a recurring schedule at a minimum quarterly, or more frequently
- Architect sections of the network to isolate critical systems, functions, or resources. Use physical and logical segmentation to prevent access to potentially sensitive systems and information. Use a DMZ to contain any internet-facing services that should not be exposed from the internal network. Configure separate virtual private cloud (VPC) instances to isolate critical cloud systems. (M1030: Network Segmentation)
    - Safeguard 12.2: Establish and Maintain a Secure Network Architecture: Establish and maintain a secure network architecture. A secure network architecture must address segmentation, least privilege, and availability, at a minimum.

- Use capabilities to detect and block conditions that may lead to or be indicative of a software exploit occurring. (M1050: Exploit Protection)
    - Safeguard 10.5: Enable Anti-Exploitation Features: Enable anti-exploitation features on enterprise assets and software, where possible, such as Microsoft® Data Execution Prevention (DEP), Windows® Defender Exploit Guard (WDEG), or Apple® System Integrity Protection (SIP) and Gatekeeper™.
- Restrict use of certain websites, block downloads/attachments, block Javascript, restrict browser extensions, etc. (M1021: Restrict Web-Based Content)
    - Safeguard 9.2: Use DNS Filtering Services: Use DNS filtering services on all enterprise assets to block access to known malicious domains.
    - Safeguard 9.3: Maintain and Enforce Network-Based URL Filters: Enforce and update network-based URL filters to limit an enterprise asset from connecting to potentially malicious or unapproved websites. Example implementations include category-based filtering, reputation-based filtering, or through the use of block lists. Enforce filters for all enterprise assets.
    - Safeguard 9.6: Block Unnecessary File Types: Block unnecessary file types attempting to enter the enterprise's email gateway.
- Remind users not to visit un-trusted websites or follow links provided by unknown or un-trusted sources. Inform and educate users regarding the threats posed by hypertext links contained in emails or attachments especially from un-trusted sources. (M1017: User Training)
    - Safeguard 14.1: Establish and Maintain a Security Awareness Program: Establish and maintain a security awareness program. The purpose of a security awareness program is to educate the enterprise's workforce on how to interact with enterprise assets and data in a secure manner. Conduct training at hire and, at a minimum, annually. Review and update content annually, or when significant enterprise changes occur that could impact this Safeguard.
    - Safeguard 14.2: Train Workforce Members to Recognize Social Engineering Attacks: Train workforce members to recognize social engineering attacks, such as phishing, pre-texting, and tailgating.

## References

https://www.jenkins.io/security/advisory/2026-10-28/

## CVEs

CVE-2026-117612
CVE-2026-117613
CVE-2026-117614
CVE-2026-117615
CVE-2026-117616
CVE-2026-117617
CVE-2026-117618
CVE-2026-117619
CVE-2026-117620
CVE-2026-117621
CVE-2026-117622
CVE-2026-117623
CVE-2026-117624

Smarttech
YOUR 24/7 SECURITY PARTNER

CVE-2026-117625
CVE-2026-117626

www.smarttech247.com