Check for updates

Publication spéciale 800-218 du NIST fre

Cadre de développement de logiciels sécurisés (SSDF), version 1.1 :

Recommandations pour atténuer le risque de vulnérabilité des logiciels

Murugiah Souppaya
Karen Scarfone
Donna Dodson

Cette publication est disponible gratuitement sur le site :
https://doi.org/10.6028/NIST.SP.800-218.fre

National Institute of Standards and Technology
U.S. Department of Commerce

Publication spéciale 800-218 du NIST fre

Cadre de développement de logiciels sécurisés (SSDF), version 1.1 :

Recommandations pour atténuer le risque de vulnérabilité des logiciels

Murugiah Souppaya
Division de la sécurité informatique
Laboratoire de technologie de l'information

Karen Scarfone
Scarfone Cybersecurity
Clifton, VA

Donna Dodson*
* *Ancienne employée du NIST ;*le NIST a réalisé tous les travaux de cette publication.

Cette publication est disponible gratuitement sur le site :
https://doi.org/10.6028/NIST.SP.800-218.fre

Février 2022

Département du commerce des États-Unis
Gina M. Raimondo, Secrétaire

Institut national des normes et de la technologie
James K. Olthoff, Exercice des fonctions et attributions non exclusives du sous-secrétaire au commerce pour les normes et la technologie et directeur de l'Institut national des normes et de la technologie

Autorité

Le NIST a élaboré cette publication conformément à ses responsabilités statutaires en vertu de la loi fédérale sur la modernisation de la sécurité de l'information (FISMA) de 2014, 44 U.S.C. § 3551 et suivants., Public Law (P.L.) 113-283. Le NIST est chargé d'élaborer des normes et des lignes directrices en matière de sécurité de l'information, y compris des exigences minimales pour les systèmes d'information fédéraux, mais ces normes et lignes directrices ne s'appliquent pas aux systèmes de sécurité nationaux sans l'approbation expresse des fonctionnaires fédéraux compétents qui exercent une autorité politique sur ces systèmes. Cette ligne directrice est conforme aux exigences de la circulaire A-130 de l'Office of Management and Budget (OMB).

Cette publication ne contient rien qui contredise les normes et les lignes directrices rendues obligatoires et contraignantes pour les agences fédérales par le Secrétaire au Commerce en vertu d'une autorité statutaire.. Les présentes lignes directrices ne doivent pas non plus être interprétées comme modifiant ou remplaçant les pouvoirs existants du secrétaire au commerce, du directeur de l'OMB ou de tout autre fonctionnaire fédéral. Les organisations non gouvernementales peuvent utiliser cette publication sur une base volontaire, et elle n'est pas soumise au droit d'auteur aux États-Unis. Le NIST apprécierait toutefois une attribution..

National Institute of Standards and Technology Special Publication 800-218 fre
Natl. Inst. Stand. Technol. Spéc. Publ. 800-218 fre, 40 pages (Février 2022)
CODEN: NSPUE2

Cette publication est disponible gratuitement sur le site :
https://doi.org/10.6028/NIST.SP.800-218.fre

Certaines entités commerciales, certains équipements ou matériaux peuvent être identifiés dans le présent document afin de décrire correctement une procédure ou un concept expérimental. Cette identification n'a pas pour but d'impliquer une recommandation ou une approbation par le NIST, ni d'impliquer que les entités, les matériaux ou l'équipement sont nécessairement les meilleurs disponibles pour l'objectif visé.

Cette publication peut référer à d'autres publications que le NIST élabore conformément à ses responsabilités statutaires. Les agences fédérales peuvent utiliser les informations contenues dans cette publication, y compris les concepts et les méthodologies, avant même l'achèvement de ces publications complémentaires. Ainsi, jusqu'à ce que chaque publication soit achevée, les exigences, lignes directrices et procédures actuelles, lorsqu'elles existent, restent en vigueur. À des fins de planification et de transition, les agences fédérales peuvent souhaiter suivre de près l'élaboration de ces nouvelles publications par le NIST.

Le NIST invite les organisations à examiner tous les projets de publication pendant les périodes de consultation publique et à lui faire part de leurs commentaires. De nombreuses publications du NIST sur la cybersécurité, autres que celles mentionnées ci-dessus, sont disponibles à l'adresse suivante https://csrc.nist.gov/publications.

Les commentaires sur cette publication doivent être adressés à : ssdf@nist.gov

Institut national des normes et de la technologie
A l'attention de : Division de la sécurité informatique, Laboratoire des technologies de l'information
100 Bureau Drive (Mail Stop 8930) Gaithersburg, MD 20899-8930

La loi sur la liberté de l'information (FOIA) peut conduire à la publication de tous les commentaires.

Traduit pour NIST par TaikaTranslations LLC sous le contrat {133ND23PNB770271}. Traduction officielle du gouvernement américain. Tous droits réservés, Secrétaire américain au commerce.

Translated for NIST by TaikaTranslations LLC under contract {133ND23PNB770271}. Official U.S. Government Translation. All rights reserved, US Secretary of Commerce.

Rapports sur la technologie des systèmes informatiques

Le Laboratoire des technologies de l'information (ITL) du National Institute of Standards and Technology (NIST) promeut l'économie américaine et le bien-être public en assurant le leadership technique de l'infrastructure nationale de mesure et de normalisation. L'ITL développe des tests, des méthodes de test, des données de référence, des mises en œuvre de démonstration de faisabilité et des analyses techniques afin de faire progresser le développement et l'utilisation productive des technologies de l'information. Les responsabilités de l'ITL comprennent l'élaboration de normes et de lignes directrices en matière de gestion, d'administration, de technique et de physique pour une sécurité et une confidentialité rentables des informations autres que celles liées à la sécurité nationale dans les systèmes d'information fédéraux. La série de publications spéciales 800 rend compte des recherches, des lignes directrices et des efforts de sensibilisation de l'ITL en matière de sécurité des systèmes d'information, ainsi que de ses activités de collaboration avec l'industrie, le gouvernement et les organisations universitaires.

Résumé

Peu de modèles de cycle de développement de logiciels (SDLC) traitent explicitement de la sécurité des logiciels en détail, de sorte que les pratiques de développement de logiciels sécurisés doivent généralement être ajoutées à chaque modèle SDLC pour garantir que le logiciel en cours de développement est bien sécurisé. Ce document recommande le cadre de développement de logiciels sécurisés (SSDF) - un ensemble de pratiques de développement de logiciels sécurisés de haut niveau qui peuvent être intégrées dans chaque mise en œuvre du SDLC. Le respect de ces pratiques devrait aider les producteurs de logiciels à réduire le nombre de vulnérabilités dans les logiciels mis sur le marché, à réduire l'impact potentiel de l'exploitation de vulnérabilités non détectées ou non traitées, et à s'attaquer aux causes profondes des vulnérabilités pour éviter qu'elles ne se reproduisent à l'avenir. Comme le cadre fournit un vocabulaire commun pour le développement de logiciels sécurisés, les acquéreurs de logiciels peuvent également l'utiliser pour favoriser la communication avec les fournisseurs dans le cadre des processus d'acquisition et d'autres activités de gestion.

Mots clés

développement de logiciels sécurisés; cadre de développement de logiciels sécurisés (SSDF); pratiques de développement de logiciels sécurisés; acquisition de logiciels; développement de logiciels; cycle de développement de logiciels (SDLC); sécurité des logiciels.

Informations sur les marques

Toutes les marques déposées ou marques commerciales appartiennent à leurs organisations respectives.

Remerciements

Les auteurs remercient toutes les organisations et les personnes qui ont contribué à cette mise à jour du SSDF. En réponse à la section 4 du décret 14028 sur l'« Amélioration de la cybersécurité nationale, » le NIST a organisé un atelier en juin 2021 et a reçu plus de 150 documents de synthèse, dont beaucoup proposaient des pratiques de développement de logiciels sécurisés, des tâches, des exemples de mise en œuvre et des références à prendre en considération pour cette mise à jour du SSDF. Les auteurs apprécient toutes ces suggestions, ainsi que les contributions des personnes qui ont pris la parole ou participé à l'atelier et qui ont fait part de leurs réflexions pendant ou après l'atelier.

En outre, les auteurs apprécient les commentaires publics soumis par des dizaines d'organisations et de personnes et souhaitent remercier Amazon Web Services, Apiiro, Blackberry, BSA | The Software Alliance, Enterprise Cloud Coalition, General Services Administration (GSA), Google, IBM, Medical Imaging & Technology Alliance (MITA), Microsoft, Oracle, Software Assurance Forum for Excellence in Code (SAFECode), Synopsys, U.S. Navy, Xoomworks, et Robert Grupe pour leurs commentaires particulièrement utiles. Des représentants de Siemens Energy et de Synopsys ont contribué à l'établissement de correspondances avec de nouvelles références.

Les auteurs remercient tous leurs collègues du NIST pour leur soutien tout au long de la mise à jour du SSDF, en particulier Curt Barker, Paul Black, Jon Boyens, Jim Foti, Barbara Guttman, Mat Heyman, Nicole Keller, Matt Scholl, Adam Sedgewick, Kevin Stine et Isabel Van Wyk.

Les auteurs souhaitent également remercier toutes les personnes et organisations qui ont fait part de leurs commentaires sur les versions préliminaires de la version originale du SSDF, notamment les Administrative Offices of the U.S. Courts, The Aerospace Corporation, BSA | The Software Alliance, Capitis Solutions, Consortium of Information & Information and Development (CIID). Aerospace Corporation, BSA | The Software Alliance, Capitis Solutions, le Consortium for Information & Software Quality (CISQ), HackerOne, The Software Alliance, HackerOne, HackerOne, The Software Alliance. Software Quality (CISQ), HackerOne, Honeycomb Secure Systems, iNovex, Ishpi Information Technologies, l'Information Security and Privacy Advisory Board (ISPAB), Juniper Networks, Microsoft, MITA, Naval Sea Systems Command (NAVSEA), NIST, Northrop Grumman, l'Office of the Undersecretary of Defense for Research and Engineering, Red Hat, SAFECode et le Software Engineering Institute (SEI).

Audience

Ce document s'adresse à deux publics principaux. Le premier concerne les producteurs de logiciels (par exemple, les vendeurs de produits commerciaux prêts à l'emploi [COTS], les développeurs de logiciels gouvernementaux prêts à l'emploi [GOTS], les développeurs de logiciels personnalisés, les équipes de développement internes), indépendamment de leur taille, de leur secteur ou de leur niveau de maturité. Le second est celui des acquéreurs de logiciels, qu'il s'agisse d'agences fédérales ou d'autres organisations. Les lecteurs du présent document ne sont pas censés être des experts en développement de logiciels sécurisés pour le comprendre, mais une telle expertise est nécessaire pour mettre en œuvre les pratiques recommandées.

Cette publication est particulièrement susceptible d'intéresser les personnes relevant des catégories de personnel et des domaines de spécialisation suivants du cadre de compétences en cybersécurité de l'Initiative nationale pour l'éducation à la cybersécurité (NICE) [SP800181] : :

- Provisionnement sécurisé (SP) : Gestion des risques (RSK), développement de logiciels (DEV), planification des exigences des systèmes (SRP), test et évaluation (TST), développement de systèmes (SYS)
- Exploitation et maintenance (OM) : Analyse des systèmes (ANA)
- Superviser et gouverner (OV) : Formation, éducation et sensibilisation (TEA) ; gestion de la cybersécurité (MGT) ; leadership exécutif dans le domaine de la cybernétique (EXL) ; gestion de programme/projet (PMA) et acquisition.
- Protéger et défendre (PR) : Réponse aux incidents (CIR), Évaluation et gestion de la vulnérabilité (VAM)
- Analyser (AN) : Analyse de la menace (TWA), Analyse de l'exploitation (EXP)

Note aux lecteurs

Nous vous encourageons à nous faire part de vos commentaires sur le SSDF à tout moment, en particulier lorsque vous le mettez en œuvre au sein de votre propre organisation et dans le cadre de vos efforts de développement de logiciels. Les contributions de divers producteurs de logiciels nous seront particulièrement utiles pour affiner et réviser le SSDF. La publication sera mise à jour périodiquement pour tenir compte de vos contributions et de vos réactions.

Si vous faites partie d'un organisme de normalisation ou d'un autre organisme qui a produit un ensemble de pratiques sécurisées et que vous souhaitez faire correspondre votre norme ou vos conseils en matière de développement de logiciels sécurisés au SSDF, veuillez nous contacter à l'adresse suivante : ssdf@nist.gov. Nous aimerions vous présenter le Programme national de références informatives en ligne (OLIR) afin que vous puissiez y soumettre votre cartographie pour compléter l'ensemble des références informatives existantes.

Avis de divulgation de brevet

AVIS : Le Laboratoire des technologies de l'information (ITL) a demandé aux détenteurs de brevets dont l'utilisation peut être nécessaire pour se conformer aux orientations ou aux exigences de la présente publication de divulguer ces brevets à l'ITL. Toutefois, les détenteurs de brevets ne sont pas obligés de répondre aux appels à brevets de l'ITL et l'ITL n'a pas entrepris de recherche de brevets afin d'identifier les éventuels brevets applicables à la présente publication.

À la date de publication, et après l'appel à identifier les revendications de brevet nécessaires pour se conformer aux orientations ou exigences de cette publication, l'ITL n'a reçu aucune notification de revendication de brevet.

ITL ne prétend pas que des licences ne sont pas nécessaires pour éviter la violation de brevets lors de l'utilisation de cette publication.

Résumé

Le présent document décrit un ensemble de pratiques fondamentales et saines pour le développement de logiciels sécurisés, appelé cadre de développement de logiciels sécurisés (SSDF). Les organisations doivent intégrer le SSDF dans leurs pratiques existantes de développement de logiciels, exprimer leurs exigences en matière de développement de logiciels sécurisés auprès de fournisseurs tiers en utilisant les conventions du SSDF, et acquérir des logiciels qui respectent les pratiques décrites dans le SSDF. L'utilisation du SSDF aide les organisations à respecter les recommandations suivantes en matière de développement de logiciels sécurisés :

- Les organisations doivent s'assurer que leur personnel, leurs processus et leur technologie sont prêts à assurer un développement sécurisé des logiciels.
- Les organisations doivent protéger tous les composants de leurs logiciels contre la falsification et l'accès non autorisé.
- Les organisations devraient produire des logiciels bien sécurisés présentant un minimum de failles de sécurité dans leurs versions.
- Les organisations doivent identifier les vulnérabilités résiduelles dans leurs versions logicielles et réagir de manière appropriée pour remédier à ces vulnérabilités et empêcher que des vulnérabilités similaires ne se produisent à l'avenir.

Le SSDF ne prescrit pas la manière de mettre en œuvre chaque pratique. L'accent est mis sur les résultats des pratiques plutôt que sur les outils, les techniques et les mécanismes. Cela signifie que les organisations de tout secteur ou de toute communauté, quelle que soit leur taille ou leur niveau de sophistication en matière de cybersécurité, peuvent utiliser le SSDF. Cela signifie que des organisations de tout secteur ou de toute communauté, peu importe leur taille ou leur niveau de maturité en cybersécurité, peuvent utiliser le SSDF.

Le SSDF ne définit qu'un sous-ensemble de haut niveau de ce que les organisations peuvent avoir à faire, de sorte que les organisations doivent consulter les références et d'autres ressources pour obtenir des informations supplémentaires sur la mise en œuvre des pratiques. Toutes les pratiques ne sont pas applicables à tous les cas d'utilisation ; les organisations doivent adopter une approche fondée sur les risques pour déterminer quelles pratiques sont pertinentes, appropriées et efficaces pour atténuer les menaces pesant sur leurs pratiques de développement de logiciels.

Les organisations peuvent indiquer comment elles traitent les clauses de la section 4 du décret présidentiel sur l'Amélioration de la cybersécurité de la nation (14028) en se référant aux pratiques et aux tâches du SSDF décrites à l'annexe A.

Table des matières

Résumé ..... **vi**
1 Introduction **1**
2 Le cadre de développement de logiciels sécurisés **5**
Références **23**
Appendix A—Le SSDF et le décret 14028 ..... **27**
Appendix B—Acronymes **28**
Appendix C—Journal des modifications **30**

Liste des tableaux

Tableau 1: Le cadre de développement de logiciels sécurisés (SSDF) version 1.1 7
Tableau 2: Pratiques du SSDF correspondant aux sous-sectionsdu décret 14028 27

1 Introduction

Un cycle de développement logiciel (SDLC)[1] est une méthodologie formelle ou informelle de conception, de création et de maintenance de logiciels (y compris le code intégré dans le matériel). Il existe de nombreux modèles de SDLC, notamment en cascade, en spirale, agile et, en particulier, agile combiné à des pratiques de développement de logiciels et d'opérations informatiques (DevOps). Peu de modèles SDLC traitent explicitement de la sécurité des logiciels en détail, de sorte que les pratiques de développement de logiciels sécurisés doivent généralement être ajoutées et intégrées à chaque modèle SDLC. Quel que soit le modèle SDLC utilisé, les pratiques de développement de logiciels sécurisés doivent être intégrées tout au long du processus pour trois raisons : réduire le nombre de vulnérabilités dans les logiciels publiés, réduire l'impact potentiel de l'exploitation de vulnérabilités non détectées ou non traitées, et s'attaquer aux causes profondes des vulnérabilités afin d'éviter qu'elles ne se reproduisent. Les vulnérabilités ne sont pas seulement des bogues dus à des défauts de codage, mais aussi des faiblesses dues à des paramètres de configuration de la sécurité, à des hypothèses de confiance incorrectes et à une analyse des risques dépassée. [IR7864]

La plupart des aspects de la sécurité peuvent être abordés à plusieurs reprises au cours du cycle de développement durable, mais en général, plus la sécurité est abordée tôt dans le cycle de développement durable, moins il faut d'efforts et de coûts pour atteindre le même niveau de sécurité. Ce principe, connu sous le nom de déplacement vers la gauche, est d'une importance cruciale quel que soit le modèle SDLC. Passer à gauche minimise la dette technique, en évitant d'avoir à corriger des failles de sécurité dès les premières étapes, que ce soit en fin de développement ou après la mise en production du logiciel. Le passage à gauche peut également déboucher sur des logiciels dont la sécurité et la résilience sont renforcées.

Il existe de nombreux documents sur les pratiques de développement de logiciels sécurisés, y compris ceux énumérés dans la section Références. Ce document n'introduit pas de nouvelles pratiques et ne définit pas de nouvelle terminologie. Il décrit plutôt un ensemble de pratiques de haut niveau basées sur des normes établies, des orientations et des documents sur les pratiques de développement de logiciels sécurisés. Ces pratiques, appelées collectivement le cadre de développement de logiciels sécurisés (SSDF), sont destinées à aider les publics cibles à atteindre les objectifs de développement de logiciels sécurisés. De nombreuses pratiques concernent directement le logiciel lui-même, tandis que d'autres l'impliquent indirectement (par exemple, la sécurisation de l'environnement de développement).

Les travaux futurs pourront développer cette publication et éventuellement couvrir des sujets tels que la manière dont le SSDF peut s'appliquer et varier pour des méthodologies de développement logiciel particulières et des pratiques associées telles que DevOps, la manière dont une organisation peut passer de ses pratiques de développement logiciel actuelles à l'incorporation des pratiques SSDF, et la manière dont le SSDF pourrait être appliqué dans le contexte des logiciels à code source ouvert. Les travaux futurs prendront probablement la forme

[1] Notez que SDLC désigne également « cycle de vie du développement du système ». Dans le présent document, le terme « SDLC » fait référence aux logiciels et non aux systèmes.

de cas d'utilisation afin que les idées soient plus facilement applicables à des types spécifiques d'environnements de développement, et ils incluront probablement une collaboration avec la communauté des logiciels libres et d'autres groupes et organisations.

Ce document identifie les pratiques de développement de logiciels sécurisés, mais ne prescrit pas la manière de les mettre en œuvre. L'accent est mis sur les résultats des pratiques à mettre en œuvre plutôt que sur les outils, les techniques et les mécanismes utilisés à cette fin. La spécification des pratiques à un niveau élevé présente les avantages suivants :

- Les organisations de tout secteur ou de toute communauté, quelle que soit leur taille ou leur niveau de sophistication en cybersécurité, peuvent l'utiliser.
- Peut s'appliquer aux logiciels développés pour soutenir les technologies de l'information (IT), les systèmes de contrôle industriel (ICS), les systèmes cyber-physiques (CPS) ou l'internet des objets (IoT).
- Peut être intégré dans n'importe quel flux de travail de développement logiciel existant et dans n'importe quelle chaîne d'outils automatisés ; ne devrait pas avoir d'incidence négative sur les organisations qui ont déjà mis en place de solides pratiques de développement logiciel sécurisé.
- Rend les pratiques largement applicables et non spécifiques à des technologies, plates-formes, langages de programmation, modèles SDLC, environnements de développement, environnements d'exploitation, outils, etc. particuliers.
- Peut aider une organisation à documenter ses pratiques actuelles de développement de logiciels sécurisés et à définir ses futures pratiques cibles dans le cadre de son processus d'amélioration continue.
- Peut aider une organisation qui utilise actuellement un modèle de développement logiciel classique à faire évoluer ses pratiques de développement logiciel sécurisé vers un modèle de développement logiciel moderne (par exemple, agile, DevOps).
- Peut aider les organisations qui achètent et utilisent des logiciels à comprendre les pratiques de développement de logiciels sécurisés employées par leurs fournisseurs.

Ce document fournit un langage commun pour décrire les pratiques fondamentales de développement de logiciels sécurisés. Cette approche est similaire à celle adoptée par le Framework for Improving Critical Infrastructure Cybersecurity, également connu sous le nom de NIST Cybersecurity Framework (cadre de cybersécurité du NIST) [NISTCSF].[2] Il n'est pas nécessaire d'être expert en développement de logiciels sécurisés pour comprendre les pratiques. Le langage commun facilite la communication sur les pratiques logicielles sécurisées entre les parties prenantes internes et externes de l'organisation, telles que

[2] Les pratiques du SSDF peuvent contribuer à soutenir les fonctions, catégories et sous-catégories du cadre de cybersécurité du NIST, mais les pratiques du SSDF n'y correspondent pas et relèvent généralement de la responsabilité de différentes parties. Les développeurs peuvent adopter les pratiques du SSDF et les résultats de leur travail pourraient aider les organisations à assurer leur sécurité opérationnelle en appui au cadre de cybersécurité.

- Propriétaires d'entreprise, développeurs de logiciels, chefs de projet, professionnels de la cybersécurité et ingénieurs d'exploitation et de plate-forme au sein d'une organisation qui ont besoin de communiquer clairement entre eux sur le développement de logiciels sécurisés.
- Les acquéreurs de logiciels, y compris les agences fédérales et d'autres organisations, qui souhaitent définir les caractéristiques requises ou souhaitées pour les logiciels dans leurs processus d'acquisition afin d'obtenir des logiciels de meilleure qualité (en particulier avec moins de vulnérabilités importantes en matière de sécurité).[3]
- Les producteurs de logiciels (par exemple, les vendeurs de produits commerciaux prêts à l'emploi, les développeurs de logiciels gouvernementaux prêts à l'emploi, les développeurs de logiciels travaillant au sein ou pour le compte d'organisations d'acquisition de logiciels) qui souhaitent intégrer des pratiques de développement de logiciels sécurisés tout au long de leur cycle de développement, faire part de leurs pratiques en matière de logiciels sécurisés à leurs clients, ou définir des exigences pour leurs fournisseurs.

Les pratiques décrites dans le présent document ne partent pas du principe que toutes les organisations ont les mêmes objectifs et priorités en matière de sécurité. Les recommandations reflètent plutôt le fait que chaque producteur de logiciel peut avoir des hypothèses de sécurité uniques et que chaque acquéreur de logiciel peut avoir des besoins et des exigences de sécurité uniques. Bien que l'objectif soit que chaque producteur de logiciels suive toutes les pratiques applicables, le degré de mise en œuvre de chaque pratique et la formalité de cette mise en œuvre varieront en fonction des hypothèses de sécurité du producteur. Les pratiques offrent une certaine souplesse aux personnes chargées de la mise en œuvre, mais elles sont également claires afin de ne pas laisser trop de place à l'interprétation.

Bien que la plupart de ces pratiques soient pertinentes pour tout effort de développement de logiciel, certaines ne le sont pas. Par exemple, si le développement d'un logiciel particulier n'implique pas l'utilisation d'un compilateur, il n'est pas nécessaire de suivre une pratique sur la configuration du compilateur pour améliorer la sécurité de l'exécutable. Certaines pratiques sont fondamentales, tandis que d'autres sont plus avancées et dépendent de certaines pratiques fondamentales déjà en place. En outre, les pratiques n'ont pas toutes la même importance dans tous les cas.

Des facteurs tels que le risque, le coût, la faisabilité et l'applicabilité doivent être pris en compte pour décider des pratiques à utiliser et du temps et des ressources à consacrer à chacune d'entre elles.[4] L'automatisabilité est également un facteur important à prendre en compte, en particulier

---

[3] Les travaux futurs pourraient fournir des conseils plus pratiques aux acquéreurs de logiciels sur la manière dont ils peuvent tirer parti du SSDF dans des cas d'utilisation spécifiques.

[4] Les organisations qui cherchent des conseils sur la manière de démarrer le développement de logiciels sécurisés peuvent consulter de nombreuses références accessibles au public, telles que « SDL That Won't Break the Bank » par Steve Lipner de SAFECode (https://i.blackhat.com/us-18/Thu-August-9/us-18-Lipner-SDL-For-The-Rest-Of-Us.pdf), « Application Software Security and the CIS Controls : A Reference Paper » by Steve Lipner and Stacy Simpson from SAFECode (https://safecode.org/resource-publications/cis-controls/), et « Simplified Implementation of the Microsoft SDL » by Microsoft (https://www.microsoft.com/en-us/download/details.aspx?id=12379).

pour la mise en œuvre de pratiques à grande échelle. Les pratiques, les tâches et les exemples de mise en œuvre représentent un point de départ à considérer ; ils sont destinés à être modifiés et personnalisés, et ne sont pas classés par ordre de priorité. Toute fréquence déclarée pour l'exécution des pratiques est fictive. L'objectif du SSDF n'est pas de créer une liste de contrôle à suivre, mais de fournir une base pour la planification et la mise en œuvre d'une approche basée sur le risque pour l'adoption de pratiques de développement de logiciels sécurisés.

La responsabilité de la mise en œuvre des pratiques peut être répartie entre différentes organisations en fonction de la fourniture du logiciel et des services (par exemple, infrastructure en tant que service, logiciel en tant que service, plateforme en tant que service, conteneur en tant que service, sans serveur). Dans ces situations, il s'agit probablement d'un modèle de responsabilité partagée impliquant les fournisseurs de plates-formes/services et l'organisation locataire qui consomme ces plates-formes/services. L'organisation locataire doit établir un accord avec les prestataires qui précise quelle partie est responsable de chaque pratique et tâche et comment chaque prestataire attestera de sa conformité à l'accord.

2 Le cadre de développement de logiciels sécurisés

Ce document définit la version 1.1 du cadre de développement sécurisé des logiciels (SSDF) avec des pratiques recommandées fondamentales, saines et sécurisées basées sur des documents de pratiques de développement sécurisé des logiciels établis. Les pratiques sont réparties en quatre groupes :

1. **Préparer l'organisation (PO) :** Les organisations doivent s'assurer que leur personnel, leurs processus et leur technologie sont prêts à assurer un développement sécurisé des logiciels au niveau de l'organisation. Pour de nombreuses organisations, certaines pratiques de PO s'appliquent également à des sous-ensembles de leur développement logiciel, tels que des groupes ou des projets de développement individuels.
2. **Protéger le logiciel (PS) :** Les organisations doivent protéger tous les composants de leurs logiciels contre la falsification et l'accès non autorisé.
3. **Produire des logiciels bien sécurisés (PW) :** Les organisations devraient produire des logiciels bien sécurisés présentant un minimum de failles de sécurité dans leurs versions.
4. **Répondre aux vulnérabilités (RV) :** Les organisations doivent identifier les vulnérabilités résiduelles dans leurs versions logicielles et réagir de manière appropriée pour remédier à ces vulnérabilités et empêcher que des vulnérabilités similaires ne se produisent à l'avenir.

Chaque définition de pratique comprend les éléments suivants :

- **Pratique :** Le nom de la pratique et un identifiant unique, suivis d'une brève explication de ce qu'est la pratique et pourquoi elle est bénéfique.
- **Tâches :** Une ou plusieurs actions qui peuvent être nécessaires pour effectuer une pratique
- **Exemples fictifs de mise en œuvre :** Un ou plusieurs exemples fictifs de types d'outils, de processus ou d'autres méthodes pouvant faciliter la mise en œuvre d'une tâche. Aucun exemple ou combinaison d'exemples n'est requis, et les exemples cités ne sont pas les seules options possibles. Certains exemples peuvent ne pas s'appliquer à certaines organisations ou situations.
- **Références :** Pointeurs vers un ou plusieurs documents sur les pratiques de développement sécurisées et leurs correspondances avec une tâche particulière. Toutes les références ne s'appliquent pas à tous les cas de développement de logiciels.

Tableau 1 définit les pratiques. Elles ne représentent qu'un sous-ensemble de ce qu'une organisation peut être amenée à faire. Le tableau présente des informations limitées par l'espace disponible ; des détails supplémentaires sur chaque pratique se trouvent dans les références. Il convient de noter que l'ordre des pratiques, des tâches et des exemples fictifs de mise en œuvre dans le tableau n'implique pas l'ordre de mise en œuvre ou l'importance relative d'une pratique, d'une tâche ou d'un exemple.

Le tableau utilise des termes tels que « données sensibles », « personne qualifiée » et « bien

sécurisé », sans les définir dans cette publication. Les organisations qui adoptent le SSDF doivent définir ces termes dans le contexte de leurs propres environnements et cas d'utilisation. Il en va de même pour les noms des environnements, tels que « développement », « construction », « mise en scène », « intégration », « test », « production » et « distribution », qui varient considérablement d'une organisation à l'autre et d'un projet à l'autre. L'énumération de vos environnements est nécessaire pour les sécuriser correctement, et surtout pour empêcher le déplacement latéral des attaquants d'un environnement à l'autre.

Tableau 1 : Le cadre de développement de logiciels sécurisés (SSDF) version 1.1

Pratiques	Tâches	Exemples de mise en œuvre théorique	Références
Préparer l'organisation (PO)			
Définir les exigences de sécurité pour le développement de logiciels (PO.1) :S'assurer que les exigences de sécurité pour le développement de logiciels sont toujours connues permet de les prendre en compte tout au long du cycle de vie du logiciel, en minimisant la duplication des efforts grâce à la collecte unique et au partage des informations sur les exigences. Il s'agit d'exigences provenant de sources internes (par exemple, les politiques de l'organisation, ses objectifs opérationnels et sa stratégie de gestion des risques) et de sources externes (par exemple, les lois et réglementations applicables).	**PO.1.1**: Identifier et documenter toutes les exigences de sécurité pour les infrastructures et les processus de développement de logiciels de l'organisation, et maintenir ces exigences au fil du temps.	**Exemple 1** : Définir des politiques pour sécuriser les infrastructures de développement logiciel et leurs composants, y compris les terminaux de développement, tout au long du cycle de développement logiciel et maintenir cette sécurité. **Exemple 2** : Définir des politiques pour sécuriser les processus de développement de logiciels tout au long du cycle de vie du logiciel et maintenir cette sécurité, y compris pour les logiciels libres et autres composants logiciels tiers utilisés par les logiciels en cours de développement. **Exemple 3** : Examiner et mettre à jour les exigences de sécurité au moins une fois par an, ou plus tôt si de nouvelles exigences émanent de sources internes ou externes, ou si un incident de sécurité majeur visant l'infrastructure de développement de logiciels s'est produit. **Exemple 4** : Informer les personnes concernées des modifications imminentes des exigences.	**BSAFSS** : SM.3, DE.1, IA.1, IA. 2 **BSIMM** : CP1.1, CP1.3, SR1.1, SR2.2, SE1.2, SE2.6 **EO14028** : 4e (ix) **IEC62443** : SM-7, SM-9 **NISTCSF** : ID.GV-3 **OWASPASVS** : 1.1.1 **OWASPMASVS** : 1,10 **OWASPSAMM** : PC1-A, PC1-B, PC2-A **PCISSLC** : 2,1, 2,2 **SCFPSSD** : Planifier la mise en œuvre et le déploiement de pratiques de développement sécurisé **SP80053** : SA-1, SA-8, SA-15, SR-3 **SP800160** : 3.1.2, 3.2.1, 3.2.2, 3.3.1, 3.4.2, 3.4.3 **SP800161** : SA-1, SA-8, SA-15, SR-3 **SP800181** : T0414 ; K0003, K0039, K0044, K0157, K0168, K0177, K0211, K0260, K0261, K0262, K0524 ; S0010, S0357, S0368 ; A0033, A0123, A0151
	PO.1.2: Identifier et documenter toutes les exigences de sécurité à respecter pour les logiciels développés par l'organisation, et maintenir ces exigences dans le temps.	**Exemple 1** : Définir des politiques qui précisent les exigences en matière d'architecture et de conception de logiciels fondées sur les risques, telles que la modularité du code pour faciliter sa réutilisation et ses mises à jour, l'isolation des composants de sécurité des autres composants pendant l'exécution, l'absence de commandes et de paramètres non documentés et la fourniture de fonctions qui aideront les acquéreurs de logiciels à assurer le déploiement, l'exploitation et la maintenance du logiciel en toute sécurité. **Exemple 2** : Définir des politiques qui précisent les exigences de sécurité pour les logiciels de l'organisation et vérifier la conformité à des moments clés du cycle de vie du logiciel (par exemple, classes de failles logicielles vérifiées par des portes, réponses aux vulnérabilités découvertes dans les logiciels diffusés). **Exemple 3** : Analyser le risque des piles technologiques applicables (par exemple, langages, environnements, modèles de déploiement) et recommander ou exiger l'utilisation de piles qui réduiront le risque par rapport à d'autres. **Exemple 4** : Établir des politiques précisant ce qu'il faut archiver pour chaque version de logiciel (par exemple, le code, les fichiers de paquets, les bibliothèques tierces, la documentation, l'inventaire des données) et la durée de conservation en fonction du modèle SDLC, de la fin de vie du logiciel et d'autres facteurs. **Exemple 5** : Veiller à ce que les politiques couvrent l'ensemble du cycle de vie des logiciels, y compris la notification aux utilisateurs de la fin imminente de l'assistance logicielle et de la date de fin de vie du logiciel. **Exemple 6** : Examiner toutes les exigences de sécurité au moins une fois par an, ou plus tôt si de nouvelles exigences émanent de sources internes ou externes, si une vulnérabilité majeure est découverte dans un logiciel publié ou si un incident de sécurité majeur visant un logiciel développé par l'organisation s'est produit. **Exemple 7** : Établir et suivre les processus de traitement des demandes de dérogation aux exigences, y compris les examens périodiques de toutes les dérogations approuvées.	**BSAFSS** : SC.1-1, SC.2, PD.1-1, PD.1-2, PD.1-3, PD.2-2, SI, PA, CS, AA, LO, EE **BSIMM** : SM1.1, SM1.4, SM2.2, CP1.1, CP1.2, CP1.3, CP2.1, CP2.3, AM1.2, SFD1.1, SFD2.1, SFD3.2, SR1.1, SR1.3, SR2.2, SR3.3, SR3.4 **EO14028** : 4e (ix) **IEC62443**: SR-3, SR-4, SR-5, SD-4 **ISO27034** : 7.3.2 **MSSDL** : 2, 5 **NISTCSF** : ID.GV-3 **OWASPMASVS** : 1,12 **OWASPSAMM** : PC1-A, PC1-B, PC2-A, PC3-A, SR1-A, SR1-B, SR2-B, SA1-B, IR1-A **PCISSLC** : 2,1, 2,2, 2,3, 3,3 **SCFPSSD** : Établir des normes et des conventions de codage **SP80053** : SA-8, SA-8 (3), SA-15, SR-3 **SP800160** : 3.1.2, 3.2.1, 3.3.1 **SP800161** : SA-8, SA-15, SR-3 **SP800181** : T0414 ; K0003, K0039, K0044, K0157, K0168, K0177, K0211, K0260, K0261, K0262, K0524 ; S0010, S0357, S0368 ; A0033, A0123, A0151
	PO.1.3: Communiquer les exigences à tous les tiers qui fourniront des composants logiciels commerciaux à l'organisation en vue de leur réutilisation par les logiciels de l'organisation. [Anciennement PW.3.1]	**Exemple 1** : Définir un ensemble d'exigences de sécurité pour les composants logiciels et l'inclure dans les documents d'acquisition, les contrats de logiciels et autres accords avec des tiers. **Exemple 2** : Définir des critères de sécurité pour la sélection des logiciels ; ces	**BSAFSS** : SM.1, SM.2, SM.2-1, SM.2-4 **BSIMM** : CP2.4, CP3.2, SR2.5, SR3.2 **EO14028** : 4e (vi), 4e (ix) **IDASOAR** : 19, 21

Pratiques	Tâches	Exemples de mise en œuvre théorique	Références
		critères peuvent inclure le programme de divulgation des vulnérabilités du tiers et les capacités de réponse aux incidents de sécurité du produit ou l'adhésion du tiers aux pratiques définies par l'organisation. **Exemple 3** : Exiger des tiers qu'ils attestent que leurs logiciels sont conformes aux exigences de sécurité de l'organisation. **Exemple 4** : Exiger des tiers qu'ils fournissent des données de provenance[5] et des mécanismes de vérification de l'intégrité pour tous les composants de leurs logiciels. **Exemple 5** : Établir et suivre des processus pour gérer les risques lorsque des composants logiciels tiers à acquérir ne répondent pas aux exigences de sécurité ; cela devrait inclure des examens périodiques de toutes les exceptions approuvées par rapport aux exigences.	**IEC62443** : SM-9, SM-10 **MSSDL** : 7 **NISTCSF** : ID.SC-3 **OWASPSAMM** : SR3-A **SCAGILE** : Tâches nécessitant l'aide d'experts en sécurité 8 **SCFPSSD** : Gérer le risque de sécurité inhérent à l'utilisation de composants tiers **SCSIC** : Contrôles de l'intégrité de l'approvisionnement des fournisseurs **SP80053** : SA-4, SA-9, SA-10, SA-10 (1), SA-15, SR-3, SR-4, SR-5 **SP800160** : 3.1.1, 3.1.2 **SP800161** : SA-4, SA-9, SA-9 (1), SA-9 (3), SA-10, SA-10 (1), SA-15, SR-3, SR-4, SR-5 **SP800181** : T0203, T0415 ; K0039 ; S0374 ; A0056, A0161
Mettre en œuvre les rôles et responsabilités (PO.2) : Veiller à ce que toutes les personnes impliquées dans le cycle de développement durable, à l'intérieur et à l'extérieur de l'organisation, soient prêtes à assumer leurs rôles et responsabilités tout au long de ce cycle.	**PO.2.1**: Créer de nouveaux rôles et modifier les responsabilités des rôles existants, si nécessaire, afin d'englober toutes les parties du cycle de développement durable. Examiner et maintenir périodiquement les rôles et responsabilités définis, et les mettre à jour si nécessaire.	**Exemple 1** : Définir les rôles et les responsabilités de tous les membres de l'équipe de développement du logiciel dans le cadre du cycle de développement du logiciel (SDLC). **Exemple 2** : Intégrer les fonctions de sécurité dans l'équipe de développement du logiciel. **Exemple 3** : Définir les rôles et les responsabilités du personnel chargé de la cybersécurité, des champions de la sécurité, des chefs de projet et des responsables, des cadres supérieurs, des développeurs de logiciels, des testeurs de logiciels, des responsables et du personnel chargés de l'assurance logicielle, des propriétaires de produits, des ingénieurs chargés des opérations et de la plateforme, et des autres personnes impliquées dans le cycle de développement logiciel (SDLC). **Exemple 4** : Procéder à un examen annuel de tous les rôles et responsabilités. **Exemple 5** : Informer les personnes concernées des changements imminents dans les rôles et les responsabilités, et confirmer que les personnes comprennent les changements et acceptent de s'y conformer. **Exemple 6** : Mettre en œuvre et utiliser des outils et des processus pour promouvoir la communication et l'engagement entre les personnes ayant des rôles et des responsabilités liés au cycle de développement durable, par exemple en créant des canaux de messagerie pour les discussions d'équipe. **Exemple 7** : Désigner un groupe de personnes ou une équipe comme propriétaire du code pour chaque projet.	**BSAFSS** : PD.2-1, PD.2-2 **BSIMM** : SM1.1, SM2.3, SM2.7, CR1.7 **EO14028** : 4e (ix) **IEC62443** : SM-2, SM-13 **NISTCSF** : ID.AM-6, ID.GV-2 **PCISSLC** : 1,2 **SCSIC** : Contrôles de l'intégrité du développement des logiciels des fournisseurs **SP80053** : SA-3 **SP800160** : 3.2.1, 3.2.4, 3.3.1 **SP800161** : SA-3 **SP800181** : K0233
	PO.2.2: Fournir une formation axée sur les rôles à l'ensemble du personnel dont les responsabilités contribuent au développement sécurisé. Examiner périodiquement les compétences du personnel et la formation basée sur les rôles, et mettre à jour la formation si nécessaire.	**Exemple 1** : Documenter les résultats souhaités de la formation pour chaque rôle. **Exemple 2** : Définir le type de formation ou de programme requis pour atteindre le résultat souhaité pour chaque rôle. **Exemple 3** : Créer un plan de formation pour chaque rôle. **Exemple 4** : Acquérir ou créer une formation pour chaque rôle ; la formation acquise peut devoir être adaptée à l'organisation. **Exemple 5** : Mesurer les résultats obtenus afin d'identifier les domaines dans lesquels des modifications de la formation pourraient être bénéfiques.	**BSAFSS** : PD.2-2 **BSIMM** : T1.1, T1.7, T1.8, T2.5, T2.8, T2.9, T3.1, T3.2, T3.4 **EO14028** : 4e (ix) **IEC62443** : SM-4 **MSSDL** : 1 **NISTCSF** : PR.AT **OWASPSAMM** : EG1-A, EG2-A **PCISSLC** : 1,3 **SCAGILE** : Tâches de sécurité opérationnelle 14, 15 ; tâches nécessitant l'aide d'experts en sécurité 1 **SCFPSSD** : Planifier la mise en œuvre et le déploiement de pratiques de développement sécurisé **SCSIC** : Contrôles de l'intégrité du développement des logiciels des fournisseurs **SP80053** : SA-8 **SP800160** : 3.2.4, 3.2.6

[5] La *provenance* est « la chronologie de l'origine, du développement, de la propriété, de l'emplacement et des modifications d'un système ou d'un composant de système et des données associées. Il peut également inclure le personnel et les processus utilisés pour interagir avec le système, le composant ou les données associées, ou pour y apporter des modifications » [SP80053].

Pratiques	Tâches	Exemples de mise en œuvre théorique	Références
			SP800161 : SA-8 **SP800181** : OV-TEA-001, OV-TEA-002 ; T0030, T0073, T0320 ; K0204, K0208, K0220, K0226, K0243, K0245, K0252 ; S0100, S0101 ; A0004, A0057
	PO.2.3: Obtenir l'engagement de la haute direction ou de l'autorité compétente de sécuriser le développement, et transmettre cet engagement à tous ceux qui ont des rôles et des responsabilités en matière de développement.	**Exemple 1** : Désigner un responsable unique ou une équipe de responsables pour l'ensemble du processus de développement de logiciels sécurisés, y compris pour la mise en production des logiciels, et déléguer les responsabilités le cas échéant. **Exemple 2** : Sensibiliser davantage les fonctionnaires chargés de délivrer les autorisations aux risques liés au développement de logiciels sans intégrer la sécurité tout au long du cycle de développement, ainsi qu'à l'atténuation des risques offerte par les pratiques de développement sécurisé. **Exemple 3** : Aider les cadres supérieurs à intégrer un soutien au développement sécurisé dans leurs communications avec le personnel dont le rôle et les responsabilités sont liés au développement. **Exemple 4** : Sensibiliser l'ensemble du personnel ayant des rôles et des responsabilités en matière de développement à l'engagement de la haute direction en faveur d'un développement sûr et à l'importance de ce développement pour l'organisation.	**BSIMM** : SM1.3, SM2.7, CP2.5 **EO14028** : 4e (ix) **NISTCSF** : ID.RM-1, ID.SC-1 **OWASPSAMM** : SM1.A **PCISSLC** : 1,1 **SP800181** : T0001, T0004

Pratiques	Tâches	Exemples de mise en œuvre théorique	Références
Mettre en œuvre des chaînes d'outils de soutien (PO.3) : Utiliser l'automatisation pour réduire l'effort humain et améliorer la précision, la reproductibilité, la facilité d'utilisation et l'exhaustivité des pratiques de sécurité tout au long du cycle de développement durable, ainsi que pour fournir un moyen de documenter et de démontrer l'utilisation de ces pratiques. Les chaînes d'outils et les outils s'utilisent à différents niveaux, que ce soit au niveau de l'organisation ou d'un projet, et peuvent se concentrer sur une partie spécifique du cycle de développement durable, comme un pipeline de construction.	**PO.3.1**: Préciser quels outils ou types d'outils doivent ou devraient être inclus dans chaque chaîne d'outils pour atténuer les risques identifiés, ainsi que la manière dont les composants de la chaîne d'outils doivent être intégrés les uns aux autres.	**Exemple 1** : Définir des catégories de chaînes d'outils et spécifier les outils ou types d'outils obligatoires à utiliser pour chaque catégorie. **Exemple 2** : Identifier les outils de sécurité à intégrer dans la chaîne d'outils du développeur. **Exemple 3** : Définir les informations à transmettre entre les outils et les formats de données à utiliser. **Exemple 4** : Évaluer les capacités de signature des outils afin de créer des enregistrements/journaux immuables à des fins d'audit au sein de la chaîne d'outils. **Exemple 5** : Utiliser une technologie automatisée pour la gestion et l'orchestration de la chaîne d'outils.	**BSIMM** : CR1.4, ST1.4, ST2.5, SE2.7 **CNCFSSCP** : Sécurisation des matériaux - vérification ; sécurisation des pipelines de construction - vérification, automatisation, authentification/accès sécurisés ; sécurisation des artefacts - vérification ; sécurisation des déploiements - vérification. **EO14028** : 4e (iii), 4e (ix) **MSSDL** : 8 **OWASPSAMM** : IR2-B, ST2-B **SCAGILE** : Tâches nécessitant l'aide d'experts en sécurité 9 **SCSIC** : Contrôles de l'intégrité de la livraison des logiciels par les fournisseurs **SP80053** : SA-15 **SP800161** : SA-15 **SP800181** : K0013, K0178
	PO.3.2: Suivre les pratiques de sécurité recommandées pour déployer, utiliser et entretenir les outils et les chaînes d'outils.	**Exemple 1** : Évaluer, sélectionner et acquérir des outils, et évaluer la sécurité de chaque outil. **Exemple 2** : Intégrer les outils à d'autres outils et aux processus et flux de travail existants en matière de développement de logiciels. **Exemple 3** : Utiliser une configuration basée sur le code pour les chaînes d'outils (par exemple, pipelines-as-code, chaînes d'outils-as-code). **Exemple 4** : Mettre en œuvre les technologies et les processus nécessaires à l'élaboration de produits reproductibles. **Exemple 5** : Mettre à jour, actualiser ou remplacer les outils en fonction des besoins afin de remédier à leurs faiblesses ou d'ajouter de nouvelles fonctionnalités. **Exemple 6** : Contrôler en permanence les outils et les journaux d'outils pour détecter d'éventuels problèmes opérationnels et de sécurité, y compris les violations de règles et les comportements anormaux. **Exemple 7** : Vérifier régulièrement l'intégrité et la provenance de chaque outil afin d'identifier les problèmes potentiels. **Exemple 8** : Voir PW.6 concernant le compilateur, l'interpréteur et les outils de construction. **Exemple 9** : Voir le PO.5 concernant la mise en œuvre et le maintien d'environnements sécurisés.	**BSAFSS** : DE. 2 **BSIMM** : SR1.1, SR1.3, SR3.4 **CNCFSSCP** : Sécurisation des pipelines de construction - vérification, automatisation, environnements contrôlés, authentification/accès sécurisés ; sécurisation des artefacts - vérification, automatisation, environnements contrôlés, chiffrement ; sécurisation des déploiements - vérification, automatisation. **EO14028** : 4e (i) (F), 4e (ii), 4e (iii), 4e (v), 4e (vi), 4e (ix) **IEC62443** : SM-7 **IR8397** : 2,2 **OWASPASVS** : 1.14.3, 1.14.4, 14.1, 14,2 **OWASPMASVS** : 7,9 **OWASPSCVS** : 3, 5 **SCAGILE** : Tâches nécessitant l'aide d'experts en sécurité 9 **SCFPSSD** : Utiliser les versions actuelles du compilateur et de la chaîne d'outils et sécuriser les options du compilateur **SCSIC** : Contrôles de l'intégrité de la livraison des logiciels par les fournisseurs **SP80053** : SA-15 **SP800161** : SA-15 **SP800181** : K0013, K0178
	PO.3.3: Configurer les outils pour qu'ils génèrent des artefacts[6] attestant qu'ils prennent en charge les pratiques de développement de logiciels sécurisés définies par l'organisation.	**Exemple 1** : Utiliser les outils existants (comme le suivi des flux de travail, le suivi des problèmes ou la cartographie de la chaîne de valeur) pour créer une piste d'audit des actions liées au développement sécurisé, afin de favoriser l'amélioration continue. **Exemple 2** : Déterminer la fréquence à laquelle les informations collectées doivent être vérifiées et mettre en œuvre les processus nécessaires. **Exemple 3** : Établir et appliquer des politiques de sécurité et de conservation des données relatives aux artefacts. **Exemple 4** : Attribuer la responsabilité de la création de tout artefact nécessaire que les outils ne peuvent pas générer.	**BSAFSS** : PD.1-5 **BSIMM** : SM1.4, SM3.4, SR1.3 **CNCFSSCP** : Sécurisation des pipelines de construction - vérification, automatisation, environnements contrôlés ; sécurisation des artefacts - vérification **EO14028** : 4e (i) (F), 4e (ii), 4e (v), 4e (ix) **IEC62443** : SM-12, SI-2 **MSSDL** : 8 **OWASPSAMM** : PC3-B **OWASPSCVS** : 3,13, 3,14 **PCISSLC** : 2,5 **SCAGILE** : Tâches nécessitant l'aide d'experts en sécurité 9 **SCSIC** : Contrôles de l'intégrité de la livraison des logiciels par les fournisseurs **SP80053** : SA-15 **SP800161** : SA-15 **SP800181** : K0013 ; T0024

[6] Un *artefact* est un « élément de preuve » [adapté de IR7692]. Les *preuves* sont « des motifs de croyance ou d'incrédulité ; des données sur lesquelles on peut fonder une preuve ou établir la vérité ou la fausseté » [SP800160]. Les artefacts constituent des enregistrements des pratiques de développement de logiciels sécurisés.

Pratiques	Tâches	Exemples de mise en œuvre théorique	Références
Définir et utiliser des critères pour les contrôles de sécurité des logiciels (PO.4) : Contribuer à garantir que les logiciels issus du SDLC répondent aux attentes de l'organisation en définissant et en utilisant des critères de contrôle de la sécurité des logiciels au cours du développement.	**PO.4.1**: Définir des critères pour les contrôles de sécurité des logiciels et assurer le suivi tout au long du cycle de développement durable.	**Exemple 1** : Veiller à ce que les critères indiquent de manière adéquate l'efficacité de la gestion des risques de sécurité. **Exemple 2** : Définir des indicateurs clés de performance (ICP), des indicateurs clés de risque (ICR), des scores de gravité des vulnérabilités et d'autres mesures pour la sécurité des logiciels. **Exemple 3** : Ajouter des critères de sécurité des logiciels aux contrôles existants (par exemple, la définition de l'action dans les méthodologies agiles du SDLC). **Exemple 4** : Examiner les artefacts générés dans le cadre du système de flux de travail pour le développement de logiciels afin de déterminer s'ils répondent aux critères. **Exemple 5 :** Enregistrer les approbations, les rejets et les demandes d'exception des contrôles de sécurité dans le cadre du flux de travail et du système de suivi. **Exemple 6** : Analyser les données collectées dans le contexte des succès et des échecs de chaque projet de développement en matière de sécurité, et utiliser les résultats pour améliorer le SDLC.	**BSAFSS** : TV.2-1, TV.5-1 **BSIMM** : SM1.4, SM2.1, SM2.2, SM2.6, SM3.3, CP2.2 **EO14028** : 4e (iv), 4e (v), 4e (ix) **IEC62443**: SI-1, SI-2, SVV-3 **ISO27034** : 7.3.5 **MSSDL** : 3 **OWASPSAMM** : PC3-A, DR3-B, IR3-B, ST3-B **PCISSLC** : 3,3 **SP80053** : SA-15, SA-15 (1) **SP800160** : 3.2.1, 3.2.5, 3.3.1 **SP800161** : SA-15, SA-15 (1) **SP800181** : K0153, K0165
	PO.4.2: Mettre en œuvre des processus, des mécanismes, etc. pour recueillir et sauvegarder les informations nécessaires à l'appui des critères.	**Exemple 1** : Utiliser la chaîne d'outils pour recueillir automatiquement des informations qui éclairent la prise de décision en matière de sécurité. **Exemple 2** : Déployer des outils supplémentaires, si nécessaire, pour faciliter la production et la collecte d'informations à l'appui des critères. **Exemple 3** : Automatiser les processus de prise de décision en utilisant les critères et revoir périodiquement ces processus. **Exemple 4** : Ne permettre qu'au personnel autorisé d'accéder aux informations recueillies et empêcher toute modification ou suppression des informations.	**BSAFSS** : PD.1-4, PD.1-5 **BSIMM** : SM1.4, SM2.1, SM2.2, SM3.4 **EO14028** : 4e (iv), 4e (v), 4e (ix) **IEC62443** : SI-1, SVV-1, SVV-2, SVV-3, SVV-4 **OWASPSAMM** : PC3-B **PCISSLC** : 2,5 **SCSIC** : Contrôles de l'intégrité de la livraison des logiciels par les fournisseurs **SP80053** : SA-15, SA-15 (1), SA-15 (11) **SP800160** : 3.2.5, 3.3.7 **SP800161** : SA-15, SA-15 (1), SA-15 (11) **SP800181** : T0349 ; K0153
Mettre en œuvre et maintenir des environnements sécurisés pour le développement de logiciels (PO.5) : Veiller à ce que tous les composants des environnements de développement de logiciels soient fortement protégés contre les menaces internes et externes afin d'empêcher la compromission des environnements ou des logiciels qui y sont développés ou maintenus. Les environnements de développement de logiciels comprennent par exemple les environnements de développement, de construction, de test et de distribution.	**PO.5.1**: Séparer et protéger chaque environnement impliqué dans le développement du logiciel.	**Exemple 1** : Utiliser une authentification multifactorielle basée sur les risques et un accès conditionnel pour chaque environnement. **Exemple 2** : Utiliser la segmentation du réseau et les contrôles d'accès pour séparer les environnements les uns des autres et des environnements de production, et pour séparer les composants les uns des autres dans chaque environnement hors production, afin de réduire les surfaces d'attaque, les mouvements latéraux des attaquants et l'escalade des privilèges/accès. **Exemple 3** : Renforcer l'authentification et limiter étroitement les connexions entrant et sortant de chaque environnement de développement logiciel, notamment en réduisant l'accès à l'internet au strict nécessaire. **Exemple 4** : Réduire au minimum l'accès humain direct aux systèmes de la chaîne d'outils, tels que les services de construction. Contrôler et auditer en permanence toutes les tentatives d'accès et toutes les utilisations d'accès privilégiés. **Exemple 5** : Réduire au minimum l'utilisation de logiciels et de services de l'environnement de production à partir d'environnements de non-production. **Exemple 6** : Enregistrer, surveiller et auditer régulièrement les relations de confiance en matière d'autorisation et d'accès entre les environnements et entre les composants de chaque environnement. **Exemple 7** : Enregistrer et surveiller en permanence les opérations et les alertes dans tous les composants de l'environnement de développement afin de détecter les tentatives d'incidents cybernétiques et les incidents réels, d'y répondre et de s'en remettre. **Exemple 8** : Configurer les contrôles de sécurité et les autres outils impliqués dans la séparation et la protection des environnements afin de générer des artefacts pour leurs activités. **Exemple 9** : Contrôler en permanence tous les logiciels déployés dans chaque	**BSAFSS** : DE.1, IA.1, IA. 2 **CNCFSSCP** : Sécuriser les pipelines de construction - environnements contrôlés **EO14028** : 4e (i) (A), 4e (i) (B), 4e (i) (C), 4e (i) (D), 4e (i) (F), 4e (ii), 4e (iii), 4e (v), 4e (vi), 4e (ix) **IEC62443** : SM-7 **NISTCSF** : PR.AC-5, PR.DS-7 **SCAGILE** : Tâches nécessitant l'aide d'experts en sécurité 11 **SCSIC** : Contrôles de l'intégrité de la livraison des logiciels par les fournisseurs **SP80053** : SA-3 (1), SA-8, SA-15 **SP800161** : SA-3, SA-8, SA-15 **SP800181** : OM-NET-001, SP-SYS-001 ; T0019, T0023, T0144, T0160, T0262, T0438, T0484, T0485, T0553 ; K0001, K0005, K0007, K0033, K0049, K0056, K0061, K0071, K0104, K0112, K0179, K0326, K0487 ; S0007, S0084, S0121 ; A0048

Pratiques	Tâches	Exemples de mise en œuvre théorique	Références
		environnement pour détecter les nouvelles vulnérabilités et réagir de manière appropriée à ces vulnérabilités en suivant une approche fondée sur les risques. **Exemple 10** : Configurer et mettre en œuvre des mesures pour sécuriser les infrastructures d'hébergement des environnements selon une architecture de confiance zéro[7].	
	PO.5.2: Sécuriser et durcir les terminaux de développement (c'est-à-dire les terminaux des concepteurs de logiciels, des développeurs, des testeurs, des constructeurs, etc.) pour qu'ils puissent effectuer des tâches liées au développement en utilisant une approche fondée sur les risques.	**Exemple 1** : Configurer chaque terminal de développement en fonction des guides de renforcement approuvés, des listes de contrôle, etc. ; par exemple, activer le chiffrement conforme aux normes FIPS de toutes les données sensibles au repos et en transit. **Exemple 2** : Configurer chaque point final de développement et les ressources de développement de manière à fournir le minimum de fonctionnalités nécessaires aux utilisateurs et aux services et à appliquer le principe du moindre privilège. **Exemple 3** : Contrôler en permanence le niveau de sécurité de tous les terminaux de développement, y compris le contrôle et l'audit de toutes les utilisations d'accès privilégiés. **Exemple 4** : Configurer les contrôles de sécurité et les autres outils impliqués dans la sécurisation et le durcissement des points de terminaison de développement afin de générer des artefacts pour leurs activités. **Exemple 5** : Exiger une authentification multifactorielle pour tous les accès aux terminaux et aux ressources de développement. **Exemple 6** : Fournir des points d'accès dédiés au développement sur les réseaux de non-production pour l'exécution de toutes les tâches liées au développement. Fournir des points d'extrémité distincts sur les réseaux de production pour toutes les autres tâches. **Exemple 7** : Configurer chaque point de développement selon une architecture de confiance zéro.	**BSAFSS** : DE.1-1, IA.1, IA.2 **EO14028** : 4e (i) (C), 4e (i)(E), 4e (i) (F), 4e (ii), 4e (iii), 4e (v), 4e (vi), 4e (ix) **IEC62443** : SM-7 **NISTCSF** : PR.AC-4, PR.AC-7, PR.IP-1, PR.IP-3, PR.IP-12, PR.PT-1, PR.PT-3, DE.CM **SCAGILE** : Tâches nécessitant l'aide d'experts en sécurité 11 **SCSIC** : Contrôles de l'intégrité de la livraison des logiciels par les fournisseurs **SP80053** : SA-15 **SP800161** : SA-15 **SP800181** : OM-ADM-001, SP-SYS-001 ; T0484, T0485, T0489, T0553 ; K0005, K0007, K0077, K0088, K0130, K0167, K0205, K0275 ; S0076, S0097, S0121, S0158 ; A0155
Logiciel de protection (PS)			
Protéger toutes les formes de code contre l'accès non autorisé et la falsification (PS.1) : Contribuer à prévenir les modifications non autorisées du code, qu'elles soient involontaires ou intentionnelles, qui pourraient contourner ou annuler les caractéristiques de sécurité prévues du logiciel. Pour le code qui n'est pas destiné à être accessible au public, cela permet d'éviter le vol du logiciel et peut rendre plus difficile ou plus long pour les attaquants de trouver des vulnérabilités dans le logiciel.	**PS.1.1**: Stocker toutes les formes de code — y compris le code source, le code exécutable et la configuration en tant que code — sur la base du principe du moindre privilège, de sorte que seuls les personnes, outils, services, etc. autorisés y aient accès.	**Exemple 1** : Stocker l'ensemble du code source et de la configuration en tant que code dans un référentiel de code et en limiter l'accès en fonction de la nature du code. Par exemple, un code source ouvert destiné à un accès public peut nécessiter la protection de son intégrité et de sa disponibilité ; d'autres codes peuvent également nécessiter la protection de leur confidentialité. **Exemple 2** : Utiliser les fonctions de contrôle de version du référentiel pour suivre toutes les modifications apportées au code et en rendre compte au compte individuel. **Exemple 3** : Utiliser la signature des livraisons pour les dépôts de code. **Exemple 4** : Demander au propriétaire du code d'examiner et d'approuver toutes les modifications apportées au code par d'autres personnes. **Exemple 5** : Utiliser le code de signature[8] pour protéger l'intégrité des exécutables. **Exemple 6** : Utiliser la cryptographie (par exemple, les hachages cryptographiques) pour protéger l'intégrité des fichiers.	**BSAFSS** : IA.1, IA.2, SM.4-1, DE.1-2 **BSIMM** : SE2.4 **CNCFSSCP** : Sécurisation du code source - vérification, automatisation, environnements contrôlés, authentification sécurisée ; sécurisation des matériaux - automatisation **EO14028** : 4e (iii), 4e (iv), 4e (ix) **IDASOAR** : Fiche d'information 25 **IEC62443** : SM-6, SM-7, SM-8 **NISTCSF** : PR.AC-4, PR.DS-6, PR.IP-3 **OWASPASVS** : 1,10, 10.3.2 **OWASPMASVS** : 7,1 **OWASPSAMM** : OE3-B **PCISSLC** : 5,1, 6,1 **SCSIC** : Contrôles de l'intégrité de la livraison des logiciels par les fournisseurs, Contrôles de l'intégrité du développement des logiciels par les fournisseurs **SP80053** : SA-10 **SP800161** : SA-8, SA-10

[7] Voir le document NIST SP 800-207, *Zero Trust Architecture,* pour de plus amples informations (https://doi.org/10.6028/NIST.SP.800-207).
[8] Pour plus d'informations sur la signature de code, voir le livre blanc du NIST sur la cybersécurité, *Security Considerations for Code Signing* (https://doi.org/10.6028/NIST.CSWP.01262018).

Pratiques	Tâches	Exemples de mise en œuvre théorique	Références
Fournir un mécanisme de vérification de l'intégrité de la version du logiciel (PS.2) : Aider les acquéreurs de logiciels à vérifier que les logiciels qu'ils obtiennent sont authentiques et intacts.	**PS.2.1**: Mettre les informations relatives à la vérification de l'intégrité des logiciels à la disposition des acquéreurs de logiciels.	**Exemple 1** : Publier les hachages cryptographiques des fichiers de diffusion sur un site web bien sécurisé. **Exemple 2** : Utiliser une autorité de certification établie pour la signature du code afin que les systèmes d'exploitation des consommateurs ou d'autres outils et services puissent confirmer la validité des signatures avant leur utilisation. **Exemple 3** : Examiner périodiquement les processus de signature du code, y compris le renouvellement, la rotation, la révocation et la protection des certificats.	**BSAFSS** : SM.4, SM.5, SM. 6 **BSIMM** : SE2.4 **CNCFSSCP** : Sécurisation des déploiements - Vérification **EO14028** : 4e (iii), 4e (ix), 4e(x) **IEC62443** : SM-6, SM-8, SUM-4 **NISTCSF** : PR.DS-6 **NISTLABEL** : 2.2.2.4 **OWASPSAMM** : OE3-B **OWASPSCVS** : 4 **PCISSLC** : 6,1, 6,2 **SCSIC** : Contrôles de l'intégrité de la livraison des logiciels par les fournisseurs **SP80053** : SA-8 **SP800161** : SA-8 **SP800181** : K0178
Archiver et protéger chaque version de logiciel (PS.3) : Conserver les versions des logiciels afin de faciliter l'identification, l'analyse et l'élimination des vulnérabilités découvertes dans les logiciels après leur publication.	**PS.3.1**: Archiver en toute sécurité les fichiers nécessaires et les données d'appui (par exemple, les informations de vérification de l'intégrité, les données de provenance) à conserver pour chaque version du logiciel.	**Exemple 1** : Stocker les fichiers de diffusion, les images associées, etc. dans des référentiels conformément à la politique établie par l'organisation. Permettre au personnel nécessaire d'y accéder en lecture seule et à personne d'autre. **Exemple 2** : Stocker et protéger les informations de vérification de l'intégrité de la libération et les données de provenance, par exemple en les conservant dans un endroit distinct des fichiers de la libération ou en signant les données.	**BSAFSS** : PD.1-5, DE.1-2, IA. 2 **CNCFSSCP** : Sécurisation des artefacts - automatisation, environnements contrôlés, chiffrement ; sécurisation des déploiements - vérification **EO14028** : 4e (iii), 4e (vi), 4e (ix), 4e(x) **IDASOAR** : 25 **IEC62443** : SM-6, SM-7 **NISTCSF** : PR.IP-4 **OWASPSCVS** : 1, 3,18, 3,19, 6,3 **PCISSLC** : 5,2, 6,1, 6,2 **SCSIC** : Contrôles de l'intégrité de la livraison des logiciels par les fournisseurs **SP80053** : SA-10, SA-15, SA-15 (11), SR-4 **SP800161** : SA-8, SA-10, SA-15 (11), SR-4
	PS.3.2: Collecter, sauvegarder, conserver et partager les données de provenance pour tous les composants de chaque version de logiciel (par exemple, dans une nomenclature logicielle [SBOM]).	**Exemple 1** : Mettre les données de provenance à la disposition des acquéreurs de logiciels conformément aux politiques de l'organisation, de préférence en utilisant des formats normalisés. **Exemple 2** : Mettre les données de provenance à la disposition des équipes d'exploitation et d'intervention de l'organisation afin de les aider à réduire les vulnérabilités des logiciels. **Exemple 3** : Protéger l'intégrité des données de provenance et fournir aux destinataires un moyen de vérifier l'intégrité des données de provenance. **Exemple 4** : Mettre à jour les données de provenance à chaque fois que l'un des composants du logiciel est mis à jour.	**BSAFSS** : SM. 2 **BSIMM** : SE3.6 **CNCFSSCP** : Sécurisation des matériaux - Vérification, automatisation **EO14028** : 4e (vi), 4e (vii), 4e (ix), 4e(x) **NTIASBOM** : Tous **OWASPSCVS** : 1,4, 2 **SCSIC** : Contrôles de l'intégrité de la livraison des logiciels par les fournisseurs **SCTPC** : MAINTAIN3 **SP80053** : SA-8, SR-3, SR-4 **SP800161** : SA-8, SR-3, SR-4

Pratiques	Tâches	Exemples de mise en œuvre théorique	Références
Produire des logiciels bien sécurisés (PW)			
Concevoir des logiciels pour répondre aux exigences de sécurité et atténuer les risques de sécurité (PW.1) : Identifier et évaluer les exigences de sécurité pour le logiciel ; déterminer les risques de sécurité auxquels le logiciel est susceptible d'être confronté pendant son fonctionnement et la manière dont la conception et l'architecture du logiciel doivent atténuer ces risques ; Justifier chaque cas où l'analyse des risques recommande d'assouplir ou de supprimer les exigences de sécurité. La prise en compte des exigences et des risques en matière de sécurité lors de la conception du logiciel (secure by design) est essentielle pour améliorer la sécurité du logiciel et contribue également à améliorer l'efficacité du développement.	**PW.1.1**: Utiliser des formes de modélisation des risques — telles que la modélisation des menaces, la modélisation des attaques ou la cartographie de la surface d'attaque — pour aider à évaluer le risque de sécurité du logiciel.	**Exemple 1** : Former l'équipe de développement (les champions de la sécurité, en particulier) ou collaborer avec un expert en modélisation des risques pour créer des modèles et analyser comment utiliser une approche basée sur les risques pour communiquer les risques et déterminer comment les traiter, y compris en mettant en œuvre des mesures d'atténuation. **Exemple 2** : Effectuer des évaluations plus rigoureuses pour les domaines à haut risque, tels que la protection des données sensibles et la sauvegarde de l'identification, de l'authentification et du contrôle d'accès, y compris la gestion des justificatifs d'identité. **Exemple 3** : Examiner les rapports de vulnérabilité et les statistiques des logiciels précédents afin d'éclairer l'évaluation des risques de sécurité. **Exemple 4** : Utiliser des méthodes de classification des données pour identifier et caractériser chaque type de données avec lesquelles le logiciel interagira.	**BSAFSS** : SC.1 **BSIMM** : AM1.2, AM1.3, AM1.5, AM2.1, AM2.2, AM2.5, AM2.6, AM2.7, SFD2.2, AA1.1, AA1.2, AA1.3, AA2.1 **EO14028** : 4e (ix) **IDASOAR**: 1 **IEC62443** : SM-4, SR-1, SR-2, SD-1 **IR8397** : 2,1 **ISO27034** : 7.3.3 **MSSDL** : 4 **NISTCSF** : ID.RA **OWASPASVS** : 1.1.2, 1,2, 1,4, 1,6, 1,8, 1,9, 1,11, 2, 3, 4, 6, 8, 9, 11, 12, 13 **OWASPMASVS** : 1,6, 1,8, 2, 3, 4, 5, 6 **OWASPSAMM** : TA1-A, TA1-B, TA3-B, DR1-A **PCISSLC** : 3,2, 3,3 **SCAGILE** : Tâches nécessitant l'aide d'experts en sécurité 3 **SCFPSSD** : Modélisation de la menace **SCTTM** : Guide complet **SP80053** : SA-8, SA-11 (2), SA-11 (6), SA-15 (5) **SP800160** : 3.3.4, 3.4.5 **SP800161** : SA-8, SA-11 (2), SA-11 (6), SA-15 (5) **SP800181** : T0038, T0062 ; K0005, K0009, K0038, K0039, K0070, K0080, K0119, K0147, K0149, K0151, K0152, K0160, K0161, K0162, K0165, K0297, K0310, K0344, K0362, K0487, K0624 ; S0006, S0009, S0022, S0078, S0171, S0229, S0248 ; A0092, A0093, A0107
	PW.1.2: Suivre et maintenir les exigences de sécurité du logiciel, les risques et les décisions de conception.	**Exemple 1** : Consigner la réponse à chaque risque, y compris la manière dont les mesures d'atténuation doivent être mises en œuvre et la justification des exceptions approuvées aux exigences de sécurité. Ajouter toute mesure d'atténuation aux exigences de sécurité du logiciel. **Exemple 2** : Conserver les enregistrements des décisions de conception, des réponses aux risques et des exceptions approuvées pour les utiliser à des fins d'audit et de maintenance tout au long du cycle de vie du logiciel. **Exemple 3** : Réévaluer périodiquement toutes les exceptions approuvées aux exigences de sécurité et mettre en œuvre les changements nécessaires.	**BSAFSS** : SC.1-1, PD.1-1 **BSIMM** : SFD3.1, SFD3.3, AA2.2, AA3.2 **EO14028** : 4e (v), 4e (ix) **IEC62443**: SD-1 **ISO27034**: 7.3.3 **MSSDL**: 4 **NISTLABEL**: 2.2.2.2 **OWASPASVS**: 1.1.3, 1.1.4 **OWASPMASVS**: 1.3, 1.6 **OWASPSAMM**: DR1-B **PCISSLC**: 3.2, 3.3 **SP80053**: SA-8, SA-10, SA-17 **SP800161** : SA-8, SA-17 **SP800181** : T0256; K0005, K0038, K0039, K0147, K0149, K0160, K0161, K0162, K0165, K0344, K0362, K0487; S0006, S0009, S0078, S0171, S0229, S0248 ; A0092, A0107
	PW.1.3: Le cas échéant, prévoir un soutien à l'utilisation de fonctions et de services de sécurité normalisés (par exemple, permettre aux logiciels de s'intégrer aux systèmes existants de gestion des journaux, de gestion des identités, de contrôle d'accès et de gestion des vulnérabilités) au lieu de créer des implémentations propriétaires de fonctions et de services de sécurité. [Anciennement PW.4.3]	**Exemple 1** : Maintenir un ou plusieurs référentiels de modules logiciels pour la prise en charge de fonctions et de services de sécurité normalisés. **Exemple 2** : Déterminer des configurations sécurisées pour les modules afin de prendre en charge les fonctions et services de sécurité normalisés, et mettre ces configurations à disposition (par exemple, sous forme de configuration en tant que code) afin que les développeurs puissent les utiliser facilement. **Exemple 3** : Définir les critères pour déterminer quelles fonctions et services de sécurité le logiciel en développement doit prendre en charge.	**BSAFSS** : SI.2-1, SI.2-2, LO. 1 **BSIMM** : SFD1.1, SFD2.1, SFD3.2, SR1.1, SR3.4 **EO14028** : 4e (ix) **IEC62443** : SD-1, SD-4 **MSSDL** : 5 **OWASPASVS** : 1.1.6 **OWASPSAMM** : SA2-A **SCFPSSD** : Normaliser la gestion des identités et des accès ; établir des exigences en matière de journalisation et des pratiques d'audit

Pratiques	Tâches	Exemples de mise en œuvre théorique	Références
Examiner la conception du logiciel pour vérifier la conformité avec les exigences de sécurité et les informations sur les risques (PW.2) : Contribuer à garantir que le logiciel répondra aux exigences de sécurité et traitera de manière satisfaisante les informations sur les risques identifiés.	**PW.2.1**: Demander à 1) une ou plusieurs personnes qualifiées non impliquées dans la conception et/ou 2) des processus automatisés instanciés dans la chaîne d'outils d'examiner la conception du logiciel pour confirmer et faire respecter qu'elle répond à toutes les exigences de sécurité et qu'elle traite de manière satisfaisante les informations sur les risques identifiées.	**Exemple 1** : Examiner la conception du logiciel pour confirmer qu'elle répond aux exigences de sécurité applicables. **Exemple 2** : Examiner les modèles de risque créés lors de la conception du logiciel pour déterminer s'ils semblent identifier les risques de manière adéquate. **Exemple 3** : Examiner la conception du logiciel pour confirmer qu'elle traite de manière satisfaisante les risques identifiés par les modèles de risque. **Exemple 4** : Demander au concepteur du logiciel de corriger les manquements aux exigences. **Exemple 5** : Modifier la conception et/ou la stratégie de réponse au risque si les exigences de sécurité ne peuvent être satisfaites. **Exemple 6** : Enregistrer les résultats des revues de conception pour servir d'artefacts (par exemple, dans la spécification du logiciel, dans le système de suivi des problèmes, dans le modèle de menace).	**BSAFSS** : TV. 3 **BSIMM** : AA1.1, AA1.2, AA1.3, AA2.1, AA3.1 **EO14028** : 4e (iv), 4e (v), 4e (ix) **IEC62443** : SM-2, SR-2, SR-5, SD-3, SD-4, SI-2 **ISO27034** : 7.3.3 **OWASPASVS** : 1.1.5 **OWASPSAMM** : DR1-A, DR1-B **PCISSLC** : 3,2 **SP800181** : T0328 ; K0038, K0039, K0070, K0080, K0119, K0152, K0153, K0161, K0165, K0172, K0297 ; S0006, S0009, S0022, S0036, S0141, S0171
Vérifier que les logiciels tiers sont conformes aux exigences de sécurité (PW.3) : *Déplacé vers PW. 4*	***PW.3.1****: Déplacé vers PO.1.3*		
	PW.3.2*: Déplacé vers PW.4.4*		
Réutiliser les logiciels existants et bien sécurisés lorsque c'est possible au lieu de dupliquer les fonctionnalités (PW.4) : Réduire les coûts de développement des logiciels, accélérer le développement des logiciels et diminuer la probabilité d'introduire des vulnérabilités de sécurité supplémentaires dans le logiciel en réutilisant des modules et des services logiciels qui ont déjà fait l'objet d'une vérification de leur niveau de sécurité. Ceci est particulièrement important pour les logiciels qui mettent en œuvre des fonctionnalités de sécurité, telles que les modules et les protocoles cryptographiques.	**PW.4.1**: Acquérir et maintenir des composants logiciels bien sécurisés (par exemple, des bibliothèques logicielles, des modules, des intergiciels, des cadres) auprès de développeurs commerciaux, de développeurs de logiciels libres et d'autres développeurs tiers pour les utiliser dans les logiciels de l'organisation.	**Exemple 1** : Examiner et évaluer les composants logiciels de tiers dans le contexte de leur utilisation prévue. En cas d'utilisation future d'un composant de manière sensiblement différente, il est nécessaire de réaliser à nouveau l'examen et l'évaluation en tenant compte de ce nouveau contexte. **Exemple 2** : Déterminer des configurations sécurisées pour les composants logiciels et les mettre à disposition (par exemple, sous forme de code de configuration) afin que les développeurs puissent facilement les utiliser. **Exemple 3** : Obtenir des informations sur la provenance (par exemple, SBOM, analyse de la composition de la source, analyse de la composition du logiciel binaire) pour chaque composant logiciel et analyser ces informations pour mieux évaluer le risque que le composant peut présenter. **Exemple 4** : Créer un ou plusieurs dépôts de logiciels pour héberger des composants à source ouverte approuvés et contrôlés. **Exemple 5** : Tenir à jour une liste des composants logiciels commerciaux approuvés par l'organisation et de leurs versions, avec leurs données de provenance. **Exemple 6** : Désigner les composants qui doivent être inclus dans le logiciel à développer. **Exemple 7** : Mettre à jour les composants logiciels déployés vers des versions plus récentes et conserver les anciennes versions jusqu'à l'achèvement de toutes les transitions. **Exemple 8** : Si l'intégrité ou la provenance des binaires acquis ne peut être confirmée, construire les binaires à partir du code source après avoir vérifié l'intégrité et la provenance du code source.	**BSAFSS** : SM. 2 **BSIMM** : SFD2.1, SFD3.2, SR2.4, SR3.1, SE3.6 **CNCFSSCP** : Sécurisation du matériel - Vérification **EO14028** : 4e (iii), 4e (vi), 4e (ix), 4e(x) **IDASOAR** : 19 **IEC62443** : SM-9, SM-10 **MSSDL** : 6 **NISTCSF** : ID.SC-2 **OWASPASVS** : 1.1.6 **OWASPSAMM** : SA1-A **OWASPSCVS** : 4 **SCSIC** : Contrôles de l'intégrité de l'approvisionnement des fournisseurs **SCTPC** : MAINTENIR **SP80053** : SA-4, SA-5, SA-8 (3), SA-10 (6), SR-3, SR-4 **SP800161** : SA-4, SA-5, SA-8 (3), SA-10 (6), SR-3, SR-4 **SP800181** : K0039
	PW.4.2: Suivre les processus SDLC pour créer et maintenir en interne des composants logiciels bien sécurisés, afin de répondre aux besoins communs de développement interne que les composants logiciels tiers ne peuvent pas mieux satisfaire.	**Exemple 1** : Suivre les pratiques de sécurité établies par l'organisation pour le développement de logiciels sécurisés lors de la création et de la maintenance des composants. **Exemple 2** : Déterminer des configurations sécurisées pour les composants logiciels et les mettre à disposition (par exemple, sous forme de code de configuration) afin que les développeurs puissent facilement les utiliser. **Exemple 3** : Maintenir un ou plusieurs dépôts de logiciels pour ces composants. **Exemple 4** : Désigner les composants qui doivent être inclus dans le logiciel à développer. **Exemple 5** : Mettre à jour les composants logiciels déployés vers des versions plus récentes, tout en conservant les anciennes versions jusqu'à réussir toutes les transitions.	**BSIMM** : SFD1.1, SFD2.1, SFD3.2, SR1.1 **EO14028** : 4e (ix) **IDASOAR** : 19 **OWASPASVS** : 1.1.6 **SCTPC** : MAINTENIR **SP80053** : SA-8 (3) **SP800161** : SA-8 (3) **SP800181** : SP-DEV-001
	PW.4.3*: Déplacé vers PW.1.3*		

Pratiques	Tâches	Exemples de mise en œuvre théorique	Références
	PW.4.4: Vérifier que les composants logiciels commerciaux, libres et tiers acquis sont conformes aux exigences, telles que définies par l'organisation, tout au long de leur cycle de vie.	**Exemple 1** : Vérifier régulièrement s'il existe des vulnérabilités connues du public dans les modules logiciels et les services que les fournisseurs n'ont pas encore corrigées. **Exemple 2** : Intégrer dans la chaîne d'outils la détection automatique des vulnérabilités connues dans les composants logiciels. **Exemple 3** : Utiliser les résultats existants des services commerciaux pour vérifier les modules et services logiciels. **Exemple 4** : S'assurer que chaque composant logiciel fait toujours l'objet d'une maintenance active et n'est pas arrivé en fin de vie ; cela devrait inclure les nouvelles vulnérabilités découvertes dans le logiciel faisant l'objet d'une mesure corrective. **Exemple 5** : Déterminer un plan d'action pour chaque composant logiciel qui n'est plus maintenu ou qui ne sera pas disponible dans un avenir proche. **Exemple 6** : Confirmer l'intégrité des composants logiciels au moyen de signatures numériques ou d'autres mécanismes. Exemple 7 : Examiner, analyser et/ou tester le code. Voir PW.7 et PW.8.	**BSAFSS** : SC.3-1, SM.2-1, SM.2-2, SM.2-3, TV.2, TV.3 **BSIMM** : CP3.2, SR2.4, SR3.1, SR3.2, SE2.4, SE3.6 **CNCFSSCP** : Sécurisation des matériaux - Vérification, automatisation **EO14028** : 4e (iii), 4e (iv), 4e (vi), 4e (ix), 4e(x) **IDASOAR**: 21 **IEC62443** : SI-1, SM-9, SM-10, DM-1 **IR8397** : 2,11 **MSSDL** : 7 **NISTCSF** : ID.SC-4, PR.DS-6 **NISTLABEL** : 2.2.2.2 **OWASPASVS** : 10, 14,2 **OWASPMASVS** : 7,5 **OWASPSAMM** : TA3-A, SR3-B **OWASPSCVS :** 4, 5, 6 **PCISSLC** : 3,2, 3,4, 4,1 **SCAGILE** : Tâches nécessitant l'aide d'experts en sécurité 8 **SCFPSSD** : Gérer le risque de sécurité inhérent à l'utilisation de composants tiers **SCSIC** : Contrôles de l'intégrité de l'approvisionnement des fournisseurs, examens par les pairs et tests de sécurité **SCTPC** : MAINTENIR, ÉVALUER **SP80053** : SA-9, SR-3, SR-4, SR-4 (3), SR-4 (4) **SP800160** : 3.1.2, 3.3.8 **SP800161** : SA-4, SA-8, SA-9, SA-9 (3), SR-3, SR-4, SR-4 (3), SR-4 (4) **SP800181** : SP-DEV-002; K0153, K0266 ; S0298
	***PW.4.5**: Déplacé vers PW.4.1 et PW.4.4*		
Créer le code source en respectant les pratiques de codage sécurisé (PW.5) : Diminuer le nombre de failles de sécurité dans le logiciel et réduire les coûts en minimisant les failles introduites lors de la création du code source qui atteignent ou dépassent les critères de gravité des failles définis par l'organisation.	**PW.5.1**: Suivre toutes les pratiques de codage sécurisé adaptées aux langages et à l'environnement de développement afin de répondre aux exigences de l'organisation.	**Exemple 1** : Valider toutes les entrées et valider et coder correctement toutes les sorties. **Exemple 2** : Éviter d'utiliser des fonctions et des appels non sûrs. **Exemple 3** : Détecter les erreurs et les traiter avec élégance. **Exemple 4** : Fournir des capacités de journalisation et de traçage. **Exemple 5** : Utiliser des environnements de développement dotés de fonctions automatisées qui encouragent ou exigent l'utilisation de pratiques de codage sécurisées avec une formation en place juste à temps. **Exemple 6** : Suivre des procédures pour assurer manuellement la conformité avec les pratiques de codage sécurisé lorsque les méthodes automatisées sont insuffisantes ou indisponibles. **Exemple 7** : Utiliser des outils (par exemple, des linters, des formateurs) pour normaliser le style et le formatage du code source. **Exemple 8** : Rechercher d'autres vulnérabilités communes aux langages et à l'environnement de développement. **Exemple 9** : Demander au développeur de réviser son propre code lisible par l'homme afin de compléter (et non de remplacer) la révision du code effectuée par d'autres personnes ou outils. Voir PW.7.	**BSAFSS** : SC.2, SC.3, LO.1, EE. 1 **BSIMM** : SR3.3, CR1.4, CR3.5 **EO14028** : 4e (iv), 4e (ix) **IDASOAR** : 2 **IEC62443** : SI-1, SI-2 **ISO27034** : 7.3.5 **MSSDL** : 9 **OWASPASVS** : 1.1.7, 1.5, 1,7, 5, 7 **OWASPMASVS** : 7,6 **SCFPSSD** : Établir des exigences en matière de journaux et des pratiques d'audit, utiliser des outils d'analyse du code pour détecter rapidement les problèmes de sécurité, manipuler les données en toute sécurité, gérer les erreurs, utiliser uniquement des fonctions sûres. **SP800181** : SP-DEV-001; T0013, T0077, T0176 ; K0009, K0016, K0039, K0070, K0140, K0624 ; S0019, S0060, S0149, S0172, S0266 **;** A0036, A0047
	***PW.5.2**: Déplacé vers PW.5.1 à titre d'exemple*		

Pratiques	Tâches	Exemples de mise en œuvre théorique	Références
Configurer les processus de compilation, d'interprétation et de construction pour améliorer la sécurité des exécutables (PW.6) : Diminuer le nombre de failles de sécurité dans le logiciel et réduire les coûts en éliminant les failles avant les tests.	**PW.6.1**: Utiliser des compilateurs, des interprètes et des outils de construction qui offrent des fonctionnalités permettant d'améliorer la sécurité des exécutables.	**Exemple 1** : Utiliser les versions les plus récentes du compilateur, de l'interpréteur et des outils de construction. **Exemple 2** : Suivre les processus de gestion des changements lors du déploiement ou de la mise à jour des compilateurs, des interprètes et des outils de construction, et vérifier tous les changements inattendus apportés aux outils. **Exemple 3** : Valider régulièrement l'authenticité et l'intégrité du compilateur, de l'interpréteur et des outils de construction. Voir PO.3.	**BSAFSS** : DE.2-1 **BSIMM** : SE2.4 **CNCFSSCP** : Sécuriser les pipelines de construction - vérification, automatisation **EO14028** : 4e (iv), 4e (ix) **IEC62443** : SI-2 **MSSDL** : 8 **SCAGILE** : Sécurité opérationnelle Tâche 3 **SCFPSSD** : Utiliser les versions actuelles du compilateur et de la chaîne d'outils et sécuriser les options du compilateur **SCSIC** : Contrôles de l'intégrité du développement des logiciels des fournisseurs **SP80053** : SA-15 **SP800161** : SA-15
	PW.6.2: Déterminer les caractéristiques du compilateur, de l'interpréteur et de l'outil de construction à utiliser et la manière dont chacune doit être configurée, puis mettre en œuvre et utiliser les configurations approuvées.	**Exemple 1** : Activer les fonctions du compilateur qui produisent des avertissements pour le code mal sécurisé pendant le processus de compilation. **Exemple 2** : Appliquer le concept de « compilation propre » en traitant tous les avertissements du compilateur comme des erreurs à éliminer, sauf ceux identifiés comme faux positifs ou non pertinents. **Exemple 3** : Réaliser toutes les constructions dans un environnement dédié et hautement contrôlé. **Exemple 4** : Activer les fonctions du compilateur qui randomisent ou obscurcissent les caractéristiques d'exécution, telles que l'utilisation de l'emplacement de la mémoire, qui seraient autrement prévisibles et donc potentiellement exploitables. **Exemple 5** : Tester pour s'assurer que les fonctionnalités fonctionnent comme prévu et qu'elles ne causent pas par inadvertance des problèmes opérationnels ou autres. **Exemple 6** :Vérifier en continu l'utilisation des configurations approuvées. **Exemple 7** : Mettre à disposition les configurations d'outils approuvées sous forme de code de configuration afin que les développeurs puissent les utiliser facilement.	**BSAFSS** : DE.2-3, DE.2-4, DE.2-5 **BSIMM** : SE2.4, SE3.2 **CNCFSSCP** : Sécuriser les pipelines de construction - vérification, automatisation **EO14028** : 4e (iv), 4e (ix) **IEC62443** : SI-2 **IR8397** : 2,5 **MSSDL** : 8 **OWASPASVS** : 14,1, 14.2.1 **OWASPMASVS** : 7,2 **PCISSLC** : 3,2 **SCAGILE** : Sécurité opérationnelle Tâche 8 **SCFPSSD** : Utiliser les versions actuelles du compilateur et de la chaîne d'outils et sécuriser les options du compilateur **SCSIC** : Contrôles de l'intégrité du développement des logiciels des fournisseurs **SP80053** : SA-15, SR-9 **SP800161** : SA-15, SR-9 **SP800181** : K0039, K0070
Examiner et/ou analyser le code lisible par l'homme afin d'identifier les vulnérabilités pour vérifier la conformité aux exigences de sécurité (PW.7) : Identifier et corriger les vulnérabilités avant la publication du logiciel pour prévenir toute exploitation. L'utilisation de méthodes automatisées réduit les efforts et les ressources nécessaires à la détection des vulnérabilités. Le code lisible par l'homme comprend le code source, les scripts et toute autre forme de code qu'une organisation considère comme lisible par l'homme.	**PW.7.1**: Décider d'utiliser l'examen du code, où une personne inspecte directement le code pour repérer les problèmes, ou l'analyse du code, avec des outils automatisés ou assistés, selon les critères définis par l'organisation.	**Exemple 1** :Respecter les politiques ou lignes directrices de l'organisation pour déterminer quand et comment effectuer l'examen du code. Il peut s'agir de code de tiers ou de modules de code réutilisables écrits en interne. **Exemple 2** :Respecter les politiques ou lignes directrices de l'organisation pour déterminer quand réaliser l'analyse du code et comment la mener. **Exemple 3** : Choisir les méthodes d'examen et/ou d'analyse du code en fonction de l'état d'avancement du logiciel.	**BSIMM** : CR1.5 **EO14028** : 4e (iv), 4e (ix) **IEC62443** : SM-5, SI-1, SVV-1 **NISTLABEL** : 2.2.2.2 **SCSIC** : Examens par les pairs et tests de sécurité **SP80053** : SA-11 **SP800161** : SA-11 **SP800181** : SP-DEV-002; K0013, K0039, K0070, K0153, K0165 ; S0174
	PW.7.2: Effectuer l'examen du code et/ou l'analyse du code sur la base des normes de codage sécurisé de l'organisation, et enregistrer et trier tous les problèmes découverts et les mesures correctives recommandées dans le flux de travail ou le système de suivi des problèmes de l'équipe de développement.	**Exemple 1** : Effectuer un examen du code par les pairs et examiner tout examen du code existant, toute analyse ou tout résultat de test dans le cadre de l'examen par les pairs. **Exemple 2** : Faire appel à des experts pour vérifier que le code ne contient pas de portes dérobées ou d'autres éléments malveillants. **Exemple 3** : Utiliser des outils d'évaluation par les pairs qui facilitent le processus d'évaluation par les pairs et documenter toutes les discussions et autres retours d'information. **Exemple 4** : Utiliser un outil d'analyse statique pour vérifier automatiquement les vulnérabilités du code et sa conformité avec les normes de codage sécurisé de l'organisation, une personne examinant les problèmes signalés par l'outil et y remédiant si nécessaire. **Exemple 5** : Utiliser des listes de contrôle pour vérifier que le code est conforme	**BSAFSS** : TV.2, PD.1-4 **BSIMM** : CR1.2, CR1.4, CR1.6, CR2.6, CR2.7, CR3.4, CR3.5 **EO14028** : 4e (iv), 4e (v), 4e (ix) **IDASOAR**: 3, 4, 5, 14, 15, 48 **IEC62443** : SI-1, SVV-1, SVV-2 **IR8397** : 2,3, 2,4 **ISO27034** : 7.3.6 **MSSDL** : 9, 10 **NISTLABEL** : 2.2.2.2 **OWASPASVS** : 1.1.7, 10 **OWASPMASVS** : 7,5 **OWASPSAMM** : IR1-B, IR2-A, IR2-B, IR3-A

Pratiques	Tâches	Exemples de mise en œuvre théorique	Références
		aux exigences. **Exemple 6** : Utiliser des outils automatisés pour identifier les pratiques logicielles dangereuses documentées et vérifiées et y remédier en permanence, au fur et à mesure que le code lisible par l'homme est vérifié dans le référentiel de code. **Exemple 7** : Identifier et documenter les causes profondes des problèmes découverts. **Exemple 8** : Documenter les enseignements tirés de l'examen et de l'analyse du code dans un wiki auquel les développeurs peuvent accéder et dans lequel ils peuvent effectuer des recherches.	**PCISSLC** : 3,2, 4,1 **SCAGILE** : Tâches de sécurité opérationnelle 4, 7 ; tâches nécessitant l'aide d'experts en sécurité 10 **SCFPSSD** : Utiliser des outils d'analyse du code pour détecter rapidement les problèmes de sécurité, utiliser des outils de test de sécurité par analyse statique, procéder à une vérification manuelle des fonctions de sécurité et des déclenchements. **SCSIC** : Examens par les pairs et tests de sécurité **SP80053** : SA-11, SA-11 (1), SA-11 (4), SA-15 (7) **SP800161** : SA-11, SA-11 (1), SA-11 (4), SA-15 (7) **SP800181** : SP-DEV-001, SP-DEV-002 ; T0013, T0111, T0176, T0267, T0516 ; K0009, K0039, K0070, K0140, K0624 ; S0019, S0060, S0078, S0137, S0149, S0167, S0174, S0242, S0266 **;** A0007, A0015, A0036, A0044, A0047
Tester le code exécutable pour identifier les vulnérabilités et vérifier la conformité aux exigences de sécurité (PW.8) : Aider à identifier les vulnérabilités afin qu'elles puissent être corrigées avant la publication du logiciel pour en empêcher l'exploitation. L'utilisation de méthodes automatisées réduit les efforts et les ressources nécessaires pour détecter les vulnérabilités et améliore la traçabilité et la reproductibilité. Le code exécutable comprend les binaires, le bytecode et le code source exécutés directement, ainsi que toute autre forme de code qu'une organisation juge exécutable.	**PW.8.1**: Décider d'effectuer des tests du code exécutable pour détecter les vulnérabilités non identifiées par les examens, analyses ou tests précédents, et déterminer quels types de tests utiliser.	**Exemple 1** : Respecter les politiques ou lignes directrices de l'organisation pour déterminer quand réaliser les tests de code et comment les conduire, par exemple dans un environnement de type « bac à sable ». Il peut s'agir de code exécutable de tiers et de modules de code exécutable réutilisables écrits en interne. **Exemple 2** : Choisir les méthodes de test en fonction de l'état d'avancement du logiciel.	**BSAFSS** : TV. 3 **BSIMM** : PT2.3 **EO14028** : 4e (ix) **IEC62443** : SVV-1, SVV-2, SVV-3, SVV-4, SVV-5 **NISTLABEL** : 2.2.2.2 **SCSIC** : Examens par les pairs et tests de sécurité **SP80053** : SA-11 **SP800161** : SA-11 **SP800181** : SP-DEV-001, SP-DEV-002 ; T0456 ; K0013, K0039, K0070, K0153, K0165, K0342, K0367, K0536, K0624 ; S0001, S0015, S0026, S0061, S0083, S0112, S0135
	PW.8.2: Étudier les tests, les concevoir, les réaliser et en documenter les résultats, y compris l'enregistrement et le triage de tous les problèmes découverts et des mesures correctives recommandées dans le flux de travail ou le système de suivi des problèmes de l'équipe de développement.	**Exemple 1** : Effectuer des tests fonctionnels robustes des dispositifs de sécurité. **Exemple 2** : Intégrer des tests de vulnérabilité dynamiques dans la suite de tests automatisés du projet. **Exemple 3** : Incorporer dans la suite de tests du projet des tests portant sur les vulnérabilités signalées précédemment, afin de s'assurer que les erreurs ne sont pas réintroduites. **Exemple 4** : Considérer les infrastructures et les piles technologiques utilisées en production lors de l'élaboration des plans de test. **Exemple 5** : Utiliser des outils de test fuzz pour détecter les problèmes liés à la gestion des entrées. **Exemple 6** : Si les ressources sont disponibles, utilisez les tests de pénétration pour simuler la manière dont un attaquant pourrait tenter de compromettre le logiciel dans des scénarios à haut risque. **Exemple 7** : Identifier et consigner les causes profondes des problèmes découverts. **Exemple 8** : Documenter les enseignements tirés des tests de code dans un wiki auquel les développeurs peuvent accéder et dans lequel ils peuvent effectuer des recherches. **Exemple 9** : Utiliser le code source, les dossiers de conception et d'autres ressources lors de l'élaboration des plans de test.	**BSAFSS** : TV.3, TV.5, PD.1-4 **BSIMM** : ST1.1, ST1.3, ST1.4, ST2.4, ST2.5, ST2.6, ST3.3, ST3.4, ST3.5, ST3.6, PT1.1, PT1.2, PT1.3, PT3.1 **EO14028** : 4e (iv), 4e (v), 4e (ix) **IDASOAR**: 7, 8, 10, 11, 38, 39, 43, 44, 48, 55, 56, 57 **IEC62443** : SM-5, SM-13, SI-1, SVV-1, SVV-2, SVV-3, SVV-4, SVV-5 **IR8397** : 2,6, 2,7, 2,8, 2,9, 2,10, 2,11 **ISO27034** : 7.3.6 **MSSDL** : 10, 11 **NISTLABEL** : 2.2.2.2 **OWASPMASVS** : 7,5 **OWASPSAMM** : ST1-A, ST1-B, ST2-A, ST2-B, ST3-A **PCISSLC** : 4,1 **SCAGILE** : Tâches de sécurité opérationnelle 10, 11 ; tâches nécessitant l'aide d'experts en sécurité 4, 5, 6, 7 **SCFPSSD** : Effectuer des tests de sécurité par analyse dynamique, Fuzz Parsers, analyse des vulnérabilités du réseau, effectuer des tests fonctionnels automatisés des fonctionnalités/mitiges de sécurité, effectuer des tests de pénétration. **SCSIC** : Examens par les pairs et tests de sécurité **SP80053 :** SA-11, SA-11 (5), SA-11 (8), SA-15 (7) **SP800161** : SA-11, SA-11 (5), SA-11 (8), SA-15 (7) **SP800181** : SP-DEV-001, SP-DEV-002 ; T0013, T0028, T0169, T0176, T0253, T0266, T0456, T0516 ; K0009, K0039, K0070, K0272, K0339, K0342, K0362, K0536, K0624 ; S0001, S0015, S0046, S0051, S0078, S0081, S0083, S0135, S0137, S0167, S0242 ; A0015

Pratiques	Tâches	Exemples de mise en œuvre théorique	Références
Configurer le logiciel pour qu'il ait des paramètres sécurisés par défaut (PW.9) : Contribuer à améliorer la sécurité du logiciel au moment de l'installation afin de réduire la probabilité que le logiciel soit déployé avec des paramètres de sécurité faibles, ce qui augmente le risque de compromission.	**PW.9.1**: Définir une base de sécurité en déterminant comment configurer chaque paramètre ayant un effet sur la sécurité ou un paramètre lié à la sécurité, de sorte que les paramètres par défaut soient sûrs et n'affaiblissent pas les fonctions de sécurité fournies par la plate-forme, l'infrastructure réseau ou les services.	**Exemple 1** : Effectuer des tests pour s'assurer que les paramètres, y compris les paramètres par défaut, fonctionnent comme prévu et n'entraînent pas par inadvertance des faiblesses en matière de sécurité, des questions opérationnelles ou d'autres problèmes.	**BSAFSS** : CF.1 **BSIMM** : SE2.2 **EO14028** : 4e (iv), 4e (ix) **IDASOAR** : 23 **IEC62443** : SD-4, SVV-1, SG-1 **ISO27034** : 7.3.5 **SCAGILE** : Tâches nécessitant l'aide d'experts en sécurité 12 **SCSIC** : Contrôles de l'intégrité de la livraison des logiciels par les fournisseurs, Contrôles de l'intégrité du développement des logiciels par les fournisseurs **SP800181** : SP-DEV-002; K0009, K0039, K0073, K0153, K0165, K0275, K0531 ; S0167
	PW.9.2: Mettre en œuvre les paramètres par défaut (ou les groupes de paramètres par défaut, le cas échéant) et documenter chaque paramètre à l'intention des administrateurs de logiciels.	**Exemple 1** : Vérifier que la configuration approuvée est en place pour le logiciel. **Exemple 2** : Documenter l'objectif de chaque paramètre, ses options, sa valeur par défaut, sa pertinence en termes de sécurité, son impact opérationnel potentiel et ses relations avec d'autres paramètres. **Exemple 3** : Programmer des mécanismes techniques pour que les administrateurs de logiciels enregistrent et évaluent la mise en œuvre de chaque paramètre. **Exemple 4** : Stocker la configuration par défaut dans un format utilisable et suivre les pratiques de contrôle des changements pour la modifier (par exemple, configuration en tant que code).	**BSAFSS** : CF.1 **BSIMM** : SE2.2 **EO14028** : 4e (iv), 4e (ix) **IDASOAR** : 23 **IEC62443** : SG-3 **OWASPSAMM** : OE1-A **PCISSLC** : 8,1, 8,2 **SCAGILE** : Tâches nécessitant l'aide d'experts en sécurité 12 **SCFPSSD** : Vérifier les configurations sécurisées et l'utilisation des mesures d'atténuation de la plate-forme **SCSIC** : Contrôles de l'intégrité de la livraison des logiciels par les fournisseurs, Contrôles de l'intégrité du développement des logiciels par les fournisseurs **SP80053** : SA-5, SA-8 (23) **SP800161** : SA-5, SA-8 (23) **SP800181** : SP-DEV-001; K0009, K0039, K0073, K0153, K0165, K0275, K0531
Répondre aux vulnérabilités (RV)			
Identifier et confirmer les vulnérabilités de manière continue (RV.1) : Contribuer à ce que les vulnérabilités soient identifiées plus rapidement afin qu'elles puissent être corrigées plus rapidement en fonction du risque, réduisant ainsi la fenêtre d'opportunité pour les attaquants.	**RV.1.1**: Recueillir des informations auprès des acquéreurs de logiciels, des utilisateurs et des sources publiques sur les vulnérabilités potentielles du logiciel et des composants tiers utilisés par le logiciel, et enquêter sur tous les rapports crédibles.	**Exemple 1** : Surveiller les bases de données sur les vulnérabilités[9], les listes de diffusion sur la sécurité et d'autres sources de rapports sur les vulnérabilités par des moyens manuels ou automatisés. **Exemple 2** : S'appuyer sur les sources de renseignements sur les menaces pour mieux comprendre l'exploitation des vulnérabilités en général. **Exemple 3** : Examiner automatiquement les données relatives à la provenance et à la composition des logiciels pour tous les composants logiciels afin d'identifier les nouvelles vulnérabilités qu'ils présentent.	**BSAFSS** : VM.1-3, VM.3 **BSIMM** : AM1.5, CMVM1.2, CMVM2.1, CMVM3.4, CMVM3.7 **CNCFSSCP** : Sécurisation du matériel - Vérification **EO14028** : 4e (iv), 4e (vi), 4e (viii), 4e (ix) **IEC62443** : DM-1, DM-2, DM-3 **ISO29147** : 6.2.1, 6.2.2, 6.2.4, 6.3, 6,5 **ISO30111** : 7.1.3 **OWASPSAMM** : IM1-A, IM2-B, EH1-B **OWASPSCVS** : 4 **PCISSLC** : 3,4, 4,1, 9,1 **SCAGILE** : Sécurité opérationnelle Tâche 5 **SCFPSSD** : Réponse aux vulnérabilités et divulgation **SCTPC** : MONITOR1 **SP80053** : SA-10, SR-3, SR-4 **SP800161** : SA-10, SR-3, SR-4 **SP800181** : K0009, K0038, K0040, K0070, K0161, K0362 ; S0078
	RV.1.2: Examiner, analyser et tester le code du logiciel pour identifier ou confirmer la présence de vulnérabilités non détectées auparavant.	**Exemple 1** : Configurer la chaîne d'outils pour effectuer des analyses et des tests de code automatisés sur une base régulière ou continue pour toutes les versions prises en charge. **Exemple 2** : Voir PW.7 et PW.8.	**BSAFSS** : VM.1-2, VM.2-1 **BSIMM** : CMVM3.1 **EO14028** : 4e (iv), 4e (vi), 4e (viii), 4e (ix) **IEC62443** : SI-1, SVV-2, SVV-3, SVV-4, DM-1, DM-2

[9] La base de données nationale sur les vulnérabilités (National Vulnerability Database - NVD) en est un exemple (https://nvd.nist.gov/).

Pratiques	Tâches	Exemples de mise en œuvre théorique	Références
			ISO27034 : 7.3.6 **ISO29147** : 6,4 **ISO30111** : 7.1.4 **PCISSLC** : 3,4, 4,1 **SCAGILE** : Tâches de sécurité opérationnelle 10, 11 **SP80053** : SA-11 **SP800161** : SA-11 **SP800181** : SP-DEV-002; K0009, K0039, K0153
	RV.1.3: Disposer d'une politique de divulgation des vulnérabilités et de remédiation, et mettre en œuvre les rôles, les responsabilités et les processus nécessaires pour soutenir cette politique.	**Exemple 1** : Mettre en place un programme de divulgation des vulnérabilités et faites en sorte qu'il soit facile pour les chercheurs en sécurité de s'informer sur votre programme et de signaler d'éventuelles vulnérabilités. **Exemple 2** : Disposer d'une équipe de réponse aux incidents de sécurité des produits (PSIRT) et de processus en place pour gérer les réponses aux rapports de vulnérabilité et aux incidents, y compris des plans de communication pour toutes les parties prenantes. **Exemple 3** : Disposer d'un manuel de réponse en matière de sécurité pour gérer une vulnérabilité générique signalée, un rapport de journées zéro, une vulnérabilité exploitée dans la nature et un incident majeur en cours impliquant plusieurs parties et des composants de logiciels libres. **Exemple 4** : Effectuer périodiquement des exercices sur les processus de réponse aux incidents de sécurité des produits.	**BSAFSS** : VM.1-1, VM.2 **BSIMM** : CMVM1.1, CMVM2.1, CMVM3.3, CMVM3.7 **EO14028** : 4e (viii), 4e (ix) **IEC62443** : DM-1, DM-2, DM-3, DM-4, DM-5 **ISO29147** : Tous **ISO30111** : Tous **MSSDL** : 12 **NISTLABEL** : 2.2.2.3 **OWASPMASVS** : 1,11 **OWASPSAMM** : IM1-A, IM1-B, IM2-A, IM2-B **PCISSLC** : 9,2, 9,3 **SCFPSSD** : Réponse aux vulnérabilités et divulgation **SP80053** : SA-15 (10) **SP800160** : 3.3.8 **SP800161** : SA-15 (10) **SP800181** : K0041, K0042, K0151, K0292, K0317 ; S0054 ; A0025 **SP800216** : Tous
Évaluer les vulnérabilités, les classer par ordre de priorité et y remédier (RV.2) : Contribuer à garantir que les vulnérabilités sont corrigées en fonction du risque afin de réduire la fenêtre d'opportunité pour les attaquants.	**RV.2.1**: Analyser chaque vulnérabilité afin de recueillir suffisamment d'informations sur le risque pour planifier sa correction ou toute autre réponse au risque.	**Exemple 1** : Utiliser le logiciel de suivi des problèmes existant pour enregistrer chaque vulnérabilité. **Exemple 2** : Effectuer des calculs de risque pour chaque vulnérabilité sur la base d'estimations de son exploitabilité, de l'impact potentiel en cas d'exploitation et de toute autre caractéristique pertinente.	**BSAFSS** : VM. 2 **BSIMM** : CMVM1.2, CMVM2.2 **EO14028** : 4e (iv), 4e (viii), 4e (ix) **IEC62443** : DM-2, DM-3 **ISO30111** : 7.1.4 **NISTLABEL** : 2.2.2.2 **PCISSLC** : 3,4, 4,2 **SCAGILE** : Tâche de sécurité opérationnelle 1, tâches nécessitant l'aide d'experts en sécurité 10 **SP80053** : SA-10, SA-15 (7) **SP800160** : 3.3.8 **SP800161** : SA-15 (7) **SP800181** : K0009, K0039, K0070, K0161, K0165 ; S0078

Pratiques	Tâches	Exemples de mise en œuvre théorique	Références
	RV.2.2: Planifier et mettre en œuvre des réponses aux risques liés aux vulnérabilités.	**Exemple 1** : Évaluer chaque vulnérabilité en fonction du risque, choisir de la corriger ou de la gérer autrement (en l'acceptant ou en la transférant), puis prioriser les actions à entreprendre. **Exemple 2** : S'il n'existe pas encore de solution permanente pour atténuer une vulnérabilité, déterminez comment la vulnérabilité peut être temporairement atténuée jusqu'à ce que la solution permanente soit disponible, et ajoutez cette solution temporaire au plan. **Exemple 3** : Élaborer et publier des avis de sécurité qui fournissent les informations nécessaires aux acquéreurs de logiciels, y compris des descriptions de la nature des vulnérabilités, de la manière de trouver des instances du logiciel vulnérable et de la manière d'y remédier (par exemple, où obtenir des correctifs et ce que les correctifs modifient dans le logiciel ; quels paramètres de configuration peuvent devoir être modifiés ; comment des solutions de contournement temporaires peuvent être mises en œuvre). **Exemple 4** : Fournir des mesures correctives aux acquéreurs par le biais d'un mécanisme de livraison automatisé et fiable. Une seule mesure corrective peut remédier à plusieurs vulnérabilités. **Exemple 5** : Mettre à jour les enregistrements des décisions de conception, des réponses aux risques et des exceptions approuvées, le cas échéant. Voir PW.1.2.	**BSAFSS** : VM.1-1, VM-2 **BSIMM** : CMVM2.1 **EO14028** : 4e (iv), 4e (vi), 4e (viii), 4e (ix) **IEC62443** : DM-4 **ISO30111** : 7.1.4, 7.1.5 **NISTLABEL** : 2.2.2.2 **PCISSLC** : 4,1, 4,2, 10,1 **SCAGILE** : Sécurité opérationnelle Tâche 2 **SCFPSSD** : Corriger la vulnérabilité, identifier les facteurs d'atténuation ou les solutions de contournement **SCTPC** : ATTÉNUER **SP80053** : SA-5, SA-10, SA-11, SA-15 (7) **SP800160** : 3.3.8 **SP800161** : SA-5, SA-8, SA-10, SA-11, SA-15 (7) **SP800181** : T0163, T0229, T0264 ; K0009, K0070
Analyser les vulnérabilités pour en identifier les causes profondes (RV.3) : Contribuer à réduire la fréquence des vulnérabilités à l'avenir.	**RV.3.1**: Analyser les vulnérabilités identifiées afin d'en déterminer les causes profondes.	**Exemple 1** : Consigner la cause première des problèmes découverts. **Exemple 2** : Consigner les enseignements tirés de l'analyse des causes profondes dans un wiki auquel les développeurs peuvent accéder et dans lequel ils peuvent effectuer des recherches.	**BSAFSS** : VM.2-1 **BSIMM** : CMVM3.1, CMVM3.2 **EO14028** : 4e (ix) **IEC62443** : DM-3 **ISO30111** : 7.1.4 **OWASPSAMM** : IM3-A **PCISSLC** : 4,2 **SCFPSSD** : Feedback sur le cycle de développement sécurisé **SP800181** : T0047, K0009, K0039, K0070, K0343
	RV.3.2: Analyser les causes profondes au fil du temps afin d'identifier des schémas, par exemple une pratique de codage sécurisée particulière qui n'est pas suivie de manière cohérente.	**Exemple 1** : Consigner les enseignements tirés de l'analyse des causes profondes dans un wiki auquel les développeurs peuvent accéder et dans lequel ils peuvent effectuer des recherches. **Exemple 2** : Ajouter des mécanismes à la chaîne d'outils pour détecter automatiquement les futures occurrences de la cause première. **Exemple 3** : Mettre à jour les processus manuels afin de détecter les futures occurrences de la cause première.	**BSAFSS** : VM.2-1, PD.1-3 **BSIMM** : CP3.3, CMVM3.2 **EO14028** : 4e (ix) **IEC62443** : DM-4 **ISO30111** : 7.1.7 **OWASPSAMM** : IM3-B **PCISSLC** : 2,6, 4,2 **SCFPSSD** : Feedback sur le cycle de développement sécurisé **SP800160** : 3.3.8 **SP800181** : T0111, K0009, K0039, K0070, K0343
	RV.3.3: Examiner le logiciel à la recherche de vulnérabilités similaires afin d'éradiquer une catégorie de vulnérabilités et de les corriger de manière proactive plutôt que d'attendre des rapports externes.	**Exemple 1** : Voir PW.7 et PW.8.	**BSAFSS** : VM. 2 **BSIMM** : CR3.3, CMVM3.1 **EO14028** : 4e (iv), 4e (viii), 4e (ix) **IEC62443** : SI-1, DM-3, DM-4 **ISO30111** : 7.1.4 **PCISSLC** : 4,2 **SP80053** : SA-11 **SP800161** : SA-11 **SP800181** : SP-DEV-001, SP-DEV-002 ; K0009, K0039, K0070

Pratiques	Tâches	Exemples de mise en œuvre théorique	Références
	RV.3.4: Examiner le processus SDLC et le mettre à jour, le cas échéant, afin d'éviter (ou de réduire la probabilité que) la cause fondamentale se reproduise dans les mises à jour du logiciel ou dans les nouveaux logiciels créés.	**Exemple 1** : Consigner les enseignements tirés de l'analyse des causes profondes dans un wiki auquel les développeurs peuvent accéder et dans lequel ils peuvent effectuer des recherches. **Exemple 2** : Planifier et mettre en œuvre des changements dans les pratiques SDLC appropriées.	**BSAFSS** : PD.1-3 **BSIMM** : CP3.3, CMVM3.2 **EO14028** : 4e (ix) **IEC62443** : DM-6 **ISO30111** : 7.1.7 **MSSDL** : 2 **PCISSLC** : 2,6, 4,2 **SCFPSSD** : Feedback sur le cycle de développement sécurisé **SP80053** : SA-15 **SP800161** : SA-15 **SP800181** : K0009, K0039, K0070

Références

[BSAFSS] BSA (2020) *The BSA Framework for Secure Software: A New Approach to Securing the Software Lifecycle, Version 1.1.* Disponible à l'adresse suivante https://www.bsa.org/files/reports/bsa_framework_secure_software_update_2020.pdf

[BSIMM] Migues S, Erlikhman E, Ewers J, Nassery K (2021) *BSIMM12 2021 Foundations Report*. Disponible à l'adresse suivante https://www.bsimm.com/content/dam/bsimm/reports/bsimm12-foundations.pdf

[CNCFSSCP] Cloud Native Computing Foundation (2021) *Software Supply Chain Best Practices.* Disponible à l'adresse suivante https://github.com/cncf/tag-security/tree/main/supply-chain-security/supply-chain-security-paper

[EO14028] Executive Order 14028 (2021) Improving the Nation's Cybersecurity. (The White House, Washington, DC), DCPD-202100401, May 12, 2021. https://www.govinfo.gov/app/details/DCPD-202100401

[IDASOAR] Hong Fong EK, Wheeler D, Henninger A (2016) State-of-the-Art Resources (SOAR) for Software Vulnerability Detection, Test, and Evaluation 2016. (Institute for Defense Analyses [IDA], Alexandria, VA), IDA Paper P-8005. Disponible à l'adresse suivante https://www.ida.org/research-and-publications/publications/all/s/st/stateoftheart-resources-soar-for-software-vulnerability-detection-test-and-evaluation-2016

[IEC62443] International Electrotechnical Commission (IEC), Security for industrial automation and control systems – Part 4-1: Secure product development lifecycle requirements, IEC 62443-4-1, 2018. Disponible à l'adresse suivante https://webstore.iec.ch/publication/33615

[IR7692] Waltermire DA, Scarfone KA, Casipe M (2011) Specification for the Open Checklist Interactive Language (OCIL) Version 2.0. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Interagency or Internal Report (IR) 7692. https://doi.org/10.6028/NIST.IR.7692

[IR7864] LeMay E, Scarfone KA, Mell PM (2012) The Common Misuse Scoring System (CMSS): Metrics for Software Feature Misuse Vulnerabilities. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Interagency or Internal Report (IR) 7864. https://doi.org/10.6028/NIST.IR.7864

[IR8397] Black P, Guttman B, Okun V (2021) Guidelines on Minimum Standards for Developer Verification of Software. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Interagency or Internal Report (IR) 8397. https://doi.org/10.6028/NIST.IR.8397

[ISO27034] International Organization for Standardization (ISO)/International Electrotechnical Commission (IEC), Information technology – Security techniques – Application security – Part 1: Overview and concepts, ISO/IEC 27034-1:2011, 2011. Disponible à l'adresse suivante https://www.iso.org/standard/44378.html

[ISO29147] International Organization for Standardization (ISO)/International Electrotechnical Commission (IEC), Information technology – Security techniques – Vulnerability disclosure, ISO/IEC 29147:2018, 2018. Disponible à l'adresse suivante https://www.iso.org/standard/72311.html

[ISO30111] International Organization for Standardization (ISO)/International Electrotechnical Commission (IEC), Information technology – Security techniques – Vulnerability handling processes, ISO/IEC 30111:2019, 2019. Disponible à l'adresse suivante https://www.iso.org/standard/69725.html

[MSSDL] Microsoft (2021) *Security Development Lifecycle*. Disponible à l'adresse suivante https://www.microsoft.com/en-us/securityengineering/sdl/

[NISTCSF] National Institute of Standards and Technology (2018) Framework for Improving Critical Infrastructure Cybersecurity, Version 1.1. (National Institute of Standards and Technology, Gaithersburg, MD). https://doi.org/10.6028/NIST.CSWP.04162018

[NISTLABEL] Ogata M, Haney J, Merkel W, Phelps A (2022) Recommended Criteria for Cybersecurity Labeling of Consumer Software. (National Institute of Standards and Technology, Gaithersburg, MD). Disponible à l'adresse suivante https://www.nist.gov/itl/executive-order-improving-nations-cybersecurity

[NTIASBOM] National Telecommunications and Information Administration (NTIA) (2021) *The Minimum Elements For a Software Bill of Materials (SBOM)*. Disponible à l'adresse suivante https://www.ntia.doc.gov/report/2021/minimum-elements-software-bill-materials-sbom

[OWASPASVS] Open Web Application Security Project (2021) *OWASP Application Security Verification Standard 4.0.3*. Disponible à l'adresse suivante https://github.com/OWASP/ASVS

[OWASPMASVS] Open Web Application Security Project (2021) *OWASP Mobile Application Security Verification Standard, Version 1.4.2*. Disponible à l'adresse suivante https://github.com/OWASP/owasp-masvs/releases

[OWASPSAMM] Open Web Application Security Project (2017) *Software Assurance Maturity Model Version 1.5*. Disponible à l'adresse suivante https://www.owasp.org/index.php/OWASP_SAMM_Project

[OWASPSCVS] Open Web Application Security Project (2020) *OWASP Software Component Verification Standard, Version 1.0*. Disponible à l'adresse suivante https://github.com/OWASP/Software-Component-Verification-Standard

[PCISSLC] Payment Card Industry (PCI) Security Standards Council (2021) *Secure Software Lifecycle (Secure SLC) Requirements and Assessment Procedures Version 1.1*. Disponible à l'adresse suivante https://www.pcisecuritystandards.org/document_library?category=sware_sec#results

[SCAGILE] Software Assurance Forum for Excellence in Code (2012) *Practical Security Stories and Security Tasks for Agile Development Environments*. Disponible à l'adresse suivante http://www.safecode.org/publication/SAFECode_Agile_Dev_Security0712.pdf

[SCFPSSD] Software Assurance Forum for Excellence in Code (2018) *Fundamental Practices for Secure Software Development: Essential Elements of a Secure Development Lifecycle Program, Third Edition*. Disponible à l'adresse suivante https://safecode.org/wp-content/uploads/2018/03/SAFECode_Fundamental_Practices_for_Secure_Software_Development_March_2018.pdf

[SCSIC] Software Assurance Forum for Excellence in Code (2010) *Software Integrity Controls: An Assurance-Based Approach to Minimizing Risks in the Software Supply Chain*. Disponible à l'adresse suivante http://www.safecode.org/publication/SAFECode_Software_Integrity_Controls0610.pdf

[SCTPC] Software Assurance Forum for Excellence in Code (2017) *Managing Security Risks Inherent in the Use of Third-Party Components*. Disponible à l'adresse suivante https://www.safecode.org/wp-content/uploads/2017/05/SAFECode_TPC_Whitepaper.pdf

[SCTTM] Software Assurance Forum for Excellence in Code (2017) *Tactical Threat Modeling*. Disponible à l'adresse suivante https://www.safecode.org/wp-content/uploads/2017/05/SAFECode_TM_Whitepaper.pdf

[SP80053] Joint Task Force (2020) Security and Privacy Controls for Information Systems and Organizations. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP) 800-53, Rev. 5. Comprend les mises à jour au 10 décembre 2020. https://doi.org/10.6028/NIST.SP.800-53r5

[SP800160] Ross R, McEvilley M, Oren J (2016) Systems Security Engineering: Considerations for a Multidisciplinary Approach in the Engineering of Trustworthy Secure Systems. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP) 800-160, Volume 1. Comprend les mises à jour au 21 mars 2018. https://doi.org/10.6028/NIST.SP.800-160v1

[SP800161] Boyens J, Smith A, Bartol N, Winkler K, Holbrook A, Fallon M (2021) Cybersecurity Supply Chain Risk Management Practices for Systems and Organizations. (National Institute of Standards and Technology, Gaithersburg, MD), Second Draft NIST Special Publication (SP) 800-161, Rev. 1. https://doi.org/10.6028/NIST.SP.800-161r1-draft2

[SP800181] Newhouse W, Keith S, Scribner B, Witte G (2017) National Initiative for Cybersecurity Education (NICE) Cybersecurity Workforce Framework. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP) 800-181. https://doi.org/10.6028/NIST.SP.800-181

[SP800216] Schaffer K, Mell P, Trinh H (2021) Recommendations for Federal Vulnerability Disclosure Guidelines. (National Institute of Standards and Technology, Gaithersburg, MD), Draft NIST Special Publication (SP) 800-216. https://doi.org/10.6028/NIST.SP.800-216-draft

Appendix A—Le SSDF et le décret 14028

Le décret présidentiel « Improving the Nation's Cybersecurity (14028) » publié le 12 mai 2021 [EO14028] a chargé plusieurs agences — dont le NIST — d'améliorer la cybersécurité par le biais de diverses initiatives liées à la sécurité et à l'intégrité de la chaîne d'approvisionnement en logiciels.

La section 4 de l'ordonnance ordonnait au NIST de solliciter l'avis du secteur privé, des universités, des agences gouvernementales et autres, et d'identifier les normes, outils, meilleures pratiques et autres lignes directrices existants ou d'en élaborer de nouveaux afin d'améliorer la sécurité de la chaîne d'approvisionnement en logiciels. Tableau 2 met en correspondance les sous-sections de la section 4e du ME avec les pratiques et les tâches du SSDF qui peuvent contribuer à traiter chaque sous-section dans le cadre d'une approche fondée sur les risques.

Tableau 2: Pratiques du SSDF correspondant aux sous-sectionsdu décret 14028

EO 14 028 Sous-section	Pratiques et tâches du SSDF
4e (i) (A)	PO.5.1
4e (i) (B)	PO.5.1
4e (i) (C)	PO.5.1, PO.5.2
4e (i) (D)	PO.5.1
4e (i)(E)	PO.5.2
4e(i)(F)	PO.3.2, PO.3.3, PO.5.1, PO.5.2
4e(ii)	PO.3.2, PO.3.3, PO.5.1, PO.5.2
4e(iii)	PO.3.1, PO.3.2, PO.5.1, PO.5.2, PS.1.1, PS.2.1, PS.3.1, PW.4.1, PW.4.4
4e(iv)	PO.4.1, PO.4.2, PS.1.1, PW.2.1, PW.4.4, PW.5.1, PW.6.1, PW.6.2, PW.7.1, PW.7.2, PW.8.2, PW.9.1, PW.9.2, RV.1.1, RV.1.2, RV.2.1, RV.2.2, RV.3.3
4e(v)	PO.3.2, PO.3.3, PO.4.1, PO.4.2, PO.5.1, PO.5.2, PW.1.2, PW.2.1, PW.7.2, PW.8.2, RV.2.2
4e(vi)	PO.1.3, PO.3.2, PO.5.1, PO.5.2, PS.3.1, PS.3.2, PW.4.1, PW.4.4, RV.1.1, RV.1.2
4e (vii)	PS.3.2
4e (viii)	RV.1.1, RV.1.2, RV.1.3, RV.2.1, RV.2.2, RV.3.3
4e (ix)	Toutes les pratiques et tâches conformes à une approche fondée sur les risques
4e(x)	PS.2.1, PS.3.1, PS.3.2, PW.4.1, PW.4.4

Parallèlement à la publication de ce document, le NIST a également publié des orientations sur la manière dont les producteurs et les acquéreurs de logiciels peuvent communiquer entre eux en ce qui concerne l'attestation de conformité à la section 4e du décret 14028.

Appendix B—Acronymes

Le document définit ci-dessous les acronymes et abréviations utilisés.

BSIMM	Construire la sécurité dans le modèle de maturité
CISQ	Consortium pour la qualité de l'information et des logiciels
CNCF	Fondation pour l'informatique en nuage
COTS	Produits commerciaux prêts à l'emploi
CPS	Système cyber-physique
DevOps	Développement et opérations
EO	Décret exécutif
GOTS	Gouvernement sur étagère
GSA	Administration des services généraux
ICS	Système de contrôle industriel
IDA	Institut d'analyse de la défense
CEI	Commission électrotechnique internationale
IdO	Internet des objets
IR	Rapport interagences ou interne
ISO	Organisation internationale de normalisation
ISPAB	Conseil consultatif de la sécurité de l'information et de la vie privée
IT	Technologie de l'information
ITL	Laboratoire de technologie de l'information
ICP	Indicateur clé de performance
IRC	Indicateur de risque clé
MITA	Alliance pour l'imagerie médicale et la technologie
NAVSEA	Commandement des systèmes navals
NICE	Initiative nationale pour l'éducation à la cybersécurité
NIST	Institut national des normes et de la technologie
NTIA	Administration nationale des télécommunications et de l'information
OLIR	Programme national de références informatives en ligne
OWASP	Projet ouvert de sécurité des applications web
PCI	Industrie des cartes de paiement
PSIRT	Équipe de réponse aux incidents de sécurité des produits
SAFECode	Forum de l'assurance logicielle pour l'excellence dans le code
SAMM	Modèle de maturité de l'assurance logicielle
SBOM	Nomenclature des logiciels
SDL	[Microsoft] Cycle de vie du développement de la sécurité
SDLC	Cycle de vie du développement logiciel
SEI	Institut du génie logiciel
CSL	Cycle de vie des logiciels

SOAR	Ressources de pointe
SSDF	Cadre de développement de logiciels sécurisés

Appendix C—Journal des modifications

Cette annexe résume les changements les plus notables apportés au SSDF depuis la publication initiale du SSDF en avril 2020.

Cette version (publiée en février 2022)

- Tâches
  - Supprimé PW.4.5 (fusionné avec PW.4.4)
- Exemples de mise en œuvre théorique
  - Ajout de nombreux exemples suggérés par les commentaires du public
  - Ajout de la mention « Exemple X » au début de chaque exemple informatif fictif
- Références
  - Ajout de EO14028, NISTLABEL, SP800161, SP800216
  - Mise à jour de BSIMM, OWASPASVS, OWASPMASVS
  - Mise à jour de NISTDVS vers IR8397
- Éditorial
  - Modifications mineures de la formulation dans l'ensemble du document
  - Ajout des définitions de « provenance », « artefact » et « preuve »

Projet publié en septembre 2021

- Pratiques
  - Ajout de PO. 5
  - Supprimé PW. 3 (fusionné avec PW.4)
- Tâches
  - Ajout de PO.1.2, PO.5.1, PO.5.2, PS.3.2, PW.1.2
  - Déplacement de PW.3.1 vers PO.1.3 ; déplacement de PW.3.2 vers PW.4.5 ; déplacement de PW.4.3 vers PW.1.3
  - Rétrogradation de l'exemple PW.5.2 en PW.5.1
- Références
  - Ajout de CNCFSSCP, IEC62443, ISO29147, ISO30111, NISTDVS, OWASPMASVS, OWASPSCVS
  - Mis à jour de BSAFSS, BSIMM, OWASPASVS, PCISSLC
  - Suppression de OWASPTEST
- Conventions du tableau SSDF
  - Suppression des identifiants pour les pratiques et tâches supprimées/modifiées (PW.3, PW.3.1, PW.3.2, PW.4.3, et PW.5.2)
  - Ajout de bordures colorées et de lignes ombrées pour chaque groupe de pratiques ; l'absence d'ombres signale les pratiques et tâches supprimées.

- Conversion du contenu d'un livre blanc en un document de la série Special Publication 800
- Ajout de l'annexe A