项目描述

软件供应链和 DevOps 安全实践

对 DevSecOps 实施基于风险的方法

Karen Scarfone

Scarfone Cybersecurity

Murugiah Souppaya

美国国家标准与技术研究所

草稿

2022 年 7 月

devsecops-nist@nist.gov

美国国家网络安全卓越中心(NCCoE)是美国国家标准与技术研究所(NIST)的一部分，是一个合作中心，在这里，行业组织、政府机构和学术机构共同工作，解决企业最紧迫的网络安全挑战。通过这种合作，美国国家网络安全卓越中心(NCCoE)开发了模块化的、适应性强的网络安全解决方案实例，演示了如何通过使用商业可用的技术来应用标准和最佳实践。欲了解更多关于美国国家网络安全卓越中心(NCCoE)的信息，请访问 https://www.nccoe.nist.gov/.要了解更多关于美国国家标准与技术研究所(NIST)的信息，请访问 https://www.nist.gov/.本文档描述了一个与许多行业部门相关的问题。美国国家网络安全卓越中心(NCCoE)网络安全专家将通过与感兴趣的社区（包括网络安全解决方案供应商）合作来应对这一挑战。由此产生的参考设计将详细介绍一种可跨多个部门合并的方法。

摘要

DevOps 将软件开发和运维结合在一起，以缩短开发周期，使组织保持敏捷性，并在利用云原生技术和实践的同时保持创新的步伐。行业和政府已经完全接受并正在迅速实施这些实践，以在运维环境中开发和部署软件，但往往没有对安全性的充分理解和考虑。此外，目前大多数软件都依赖于一个或多个第三方组件，但组织通常对这些组件是如何开发、集成和部署的，以及用于确保组件安全性的实践很少或者几乎没有了解。为了帮助提高 DevOps 实践的安全性，美国国家网络安全卓越中心(NCCoE)正在计划一个 DevSecOps 项目，该项目最初将专注于开发和记录一种基于风险的应用方法和建议，用于符合安全软件开发框架（SSDF）、网络安全供应链风险管理（C-SCRM）和其他美国国家标准与技术研究所(NIST)、政府和行业指南的安全 DevOps 和软件供应链实践。该项目将把这些 DevSecOps 实践应用于概念验证用例场景中，每个场景都将特定于技术、编程语言和行业部门。商业和开源技术都将被用于演示这些用例。这个项目将会产生一个免费的美国国家标准与技术研究所(NIST)网络安全实践指南。

关键字

云原生技术；网络安全供应链风险管理；开发行动；DevSecOps；安全软件开发；安全软件开发框架(SSDF)；供应链安全

免责声明

某些商业实体、设备、产品或材料可以在本文件中进行标识，以充分描述一个实验程序或概念。此种类标识并不意味着美国国家标准与技术研究所(NIST)或美国国家网络安全卓越中心(NCCoE)的推荐或认可，也不意味着这些实体、设备、产品或材料必然是实现该目的的最佳选择。

对美国国家网络安全卓越中心(NCCoE)文件的评论

鼓励各组织在公众评论期间审查所有出版物草案并提供反馈。所有来自美国国家标准与技术研究所(NIST)的国家网络安全卓越中心的出版物均可在 https://www.nccoe.nist.gov/.

对本出版物的评论意见可提交给 devsecops-nist@nist.gov

公众评议期：2022 年 7 月 21 日至 2022 年 8 月 22 日

目录

1 执行摘要

目的：

DevOps 将软件开发和运维结合在一起，以缩短开发周期，使组织保持敏捷性，并在利用云原生技术和实践的同时保持创新的步伐。行业和政府已经完全接受并正在迅速实施这些实践，以在运维环境中开发和部署软件，但往往没有对安全性的充分理解和考虑。DevSecOps 通过集成安全实践并在整个过程（包括软件开发、构建、打包、分发和部署）中自动生成安全和合规工件，帮助确保安全性作为所有 DevOps 实践的一部分得到解决。这一点很重要，原因有几个原因，包括：

◎ 减少已发布软件中的漏洞、恶意代码和其他安全问题，而不减缓代码的生产和发布；

◎ 在整个软件生命周期中减轻漏洞利用的潜在影响，包括在动态托管平台上开发、构建、打包、分发、部署和执行软件的时间；

◎ 解决漏洞的根本问题，以防止复发，例如加强工具链中的测试工具和方法，以及改进开发代码和运维托管平台的实践；

◎ 减少开发、运维和安全团队之间的摩擦，以保持在利用现代和创新技术的同时，实现支持组织使命所需的速度和敏捷性。

越来越多的人认识到，DevSecOps 也应该包括软件供应链安全。如今，大多数软件依赖于一个或多个第三方组件，但组织通常很少或根本没有对这些软件组件是如何开发、集成和部署以及用于确保组件安全性的实践有可见性和理解。DevSecOps 的实践可以帮助识别、评估和减轻软件供应链的网络安全风险。[1]

本文件定义了一个国家网络安全卓越中心(NCCoE)项目，我们正在对此寻求反馈。本项目的重点是开发和记录适用于 DevSecOps 实践的基于风险的方法和建议。就本项目而言，术语“DevSecOps”是指将安全团队开发的安全实践集成到现有管道(例如，持续集成/连续交付[CI/CD])和由开发人员使用并由运维团队管理的现有工具链中。美国国家标准与技术研究所(NIST)为这个项目提出的方法与美国国家标准与技术研究所(NIST)安全软件开发框架(SSDF)所使用的方法类似[2]以及美国国家标准与技术研究所(NIST)的网络安全框架[3].该项目旨在帮助组织能够以云原生的方式维护软件交付的速度和质量，并利用自动化工具。该项目还将确定如何将来自美国国家标准与技术研究所(NIST) 安全软件开发框架（SSDF）的实践和任务作为 DevSecOps 方法的一部分来实现。

该项目的目标是制定实用的和可操作的指导方针，以便有意义地将安全实践集成到开发方法中。工业、政府和其他组织可以在选择和实施 DevSecOps 实践时应用这些指导方针，以提高他们开发和运维的软件的安全性。反过来，这将提高使用该软件的组织的安全性，以及整个软件供应链的安全性。此外，该项目旨在演示组织如何生成工件作为其 DevSecOps 实践的副产品，以支持和告知组织的自我认证和声明，以符合适用的美国国家标准与技术研究所(NIST)和行业推荐的安全软件开发和网络安全供应链风险管理实践。

该项目还将努力展示使用当前和新兴的安全开发框架、实践和工具来应对网络安全挑战。在项目过程中吸取的经验教训将与安全和软件开发社区分享，以为安全开发框架、实践和工具的改进提供信息。所吸取的教训也将与标准开发组织共享，以告知他们与 DevSecOps 相关的工作。

该项目将产生一个公开的美国国家标准与技术研究所(NIST)网络安全实践指南，一个详细的实施指南，说明实施网络安全参考设计所需的实际步骤，以应对这一挑战。

范围

该项目将把 DevSecOps 实践应用于多个概念验证用例场景中，每个用例场景都涉及不同的技术、编程语言、行业部门等。美国国家网络安全卓越中心(NCCoE)项目将使用商业和开源技术来演示这些用例。其目的是演示 DevSecOps 的实践，它将适用于所有规模和所有部门的组织，以及信息技术(IT)、运维技术(OT)、物联网(IoT)和其他技术类型的开发。这个项目将不会专注于任何特定技术类型的开发。

作为该项目的一部分，美国国家标准与技术研究所(NIST)将从现有的指导和实践出版物中汇集并规范关于 DevSecOps 实践的内容。本内容将作为该项目的美国国家标准与技术研究所(NIST)网络安全

实践指南的一部分发布，并将与用例实施并行起草和修订。它将提供基本的 DevSecOps 概念的定义，以便开发人员、安全专业人员和运维人员都可以对它们有相同的共同理解。此外，它还将记录组织构建成功的 DevSecOps 实践所需要的关键元素，从改变组织文化到将安全实践自动化为现有的开发管道和工具链，以支持持续授权(ATO)的概念。该指南还将为所有组织提供一种记录其当前的 DevSecOps 实践的方法，并将其未来的目标实践定义为其持续改进过程的一部分。指南中的建议和实践将为选择采用它们的组织在实施时提供灵活性和可定制性。

选择与 DevOps 和供应链安全最密切相关的选定美国国家标准与技术研究所(NIST)指南，如美国国家标准与技术研究所(NIST)特别出版物(SP)800-218[2], SP 800-190[4], 和 SP800-161[1], 将被用于用例实现，并可能在项目过程中根据从实现中获得的经验教训进行更新。美国国家标准与技术研究所(NIST)和其他公司已经有了许多现有的安全指导和实践出版物，但它们尚未被纳入 DevOps 或 DevSecOps 的上下文中。行业、标准开发组织、政府机构和其他机构已经在执行 DevSecOps。将利用他们的努力，提供一套由社区发展的推荐做法。更新受影响的美国国家标准与技术研究所(NIST)出版物，使其反映 DevSecOps 原则，也将帮助组织更好地利用他们的建议。

假设和挑战

假设读者能够理解基本的 DevOps 和安全的软件开发概念。

背景

一个软件开发生命周期(SDLC)[1] 是用于设计、创建和维护软件（包括内置到硬件中的代码）的一种正式的或非正式的方法。软件开发生命周期（SDLC）有许多模型，包括瀑布式、螺旋式、敏捷式，特别是与软件开发和 IT 运维(DevOps)实践相结合的敏捷式。很少有软件开发生命周期（SDLC）模型明确地详细说明了软件安全性，因此通常需要将安全的软件开发实践添加到每个软件开发生命周期（SDLC）模型中并集成到其中。无论使用哪种软件开发生命周期（SDLC）模型，安全软件开发实践都应该贯穿其中，原因有三：减少发布的漏洞的数量，减少利用未被发现的潜在影响或未解决的漏洞，并解决漏洞的根本问题，防止复发。漏洞不仅包括由编码缺陷引起的错误，还包括由安全配置设置、不正确的信任假设和过时或不正确的风险分析造成的弱点。[5]

安全的大多数方面都可以在软件开发生命周期（SDLC）中的多个地方进行解决，通常在成本、有效性和集成的易用性方面存在一些差异。然而，一般来说，在软件开发生命周期（SDLC）中，解决安全性的时间越早，要达到相同的安全级别，最终需要的努力和成本就越低。无论软件开发生命周期（SDLC）模型如何，这个原则被称为左移，都是至关重要的。向左移动可以减少任何技术债务，这需要在开发后期或软件生产后纠正早期安全缺陷。左移也会导致软件具有更强的安全性。

对于今天的软件，实现安全实践的责任通常是基于交付机制（例如，基础设施即服务、软件即服务、平台即服务、容器即服务、无服务器）。在这些情况下，它很可能遵循一个共享的责任模式，包括平台/服务提供商和使用这些平台/服务的租户组织。双方将需要根据组织定义的政策、法规和授权，商定需要执行哪些安全实践，哪一方对每个实践负责，以及各方将如何证明其符合协议。

当今软件的另一个方面是，它经常使用由其他组织开发的一个或多个软件组件。其中一些组件也可能使用来自其他组织的组件，等等。作为网络安全供应链风险管理(C-SCRM)的一部分，管理来自第三方软件组件的网络安全风险，包括识别、评估、选择和实施流程和减轻控制。这种风险管理可以通过其自动化能力在很大程度上集成到 DevSecOps 中。

[1] 请注意，SDLC 也被广泛用于“系统开发生命周期”。本文档中“SDLC”的所有使用都是引用软件，而不是系统。

2 场景

我们将为这个项目考虑的用例场景。

场景 1：免费和开源软件(FOSS)开发

这个场景涉及一个小型自由和开源软件社区，该社区希望提高其软件的安全性。自由/开源软件社区都是以志愿者为基础的。他们还希望为其他想要使用该软件的人提供更好的安全透明度，包括来源信息和确认软件完整性的机制。这个社区已经使用了基于云的、公开可访问的软件开发、打包和分发库。该软件本身依赖于来自其他社区的多个开源组件。

场景 2：商业现货软件开发

此场景涉及一个中型大型组织，该组织为其全球客户提供现有的基于云的应用程序。该组织正在积极地开发、维护和支持该应用程序，它利用了多个商业和开源组件。该应用程序的生产环境是在公共云中进行的，并且是基于微服务的。开发和构建环境、版本控制系统、代码存储库和工具链的其他部分分散在私有云和软件即服务(SaaS)托管的应用程序中。在此场景中，组织希望确保其 DevSecOps 方法解决安全软件开发框架（SSDF）云环境中的所有适用实践，以及生成工件来支持和通知其自我认证和声明符合适用的美国国家标准与技术研究所(NIST)和行业推荐的安全软件开发和网络安全供应链风险管理实践。

对于每个场景，我们都将执行一个或多个构建实现。每个构建实现都将与其他构建实现有显著的不同，例如使用不同的技术堆栈和编程语言。每个构建实现都将在可行的程度内依赖于自动化，例如使用现有功能或在现有平台和工具中添加自动化功能。此外，每个构建实现都将解决整个软件开发生命周期中的安全性问题，以包括开发人员、集成、构建、部署和分发系统的安全性。

3 高级架构

组件清单

开发和托管环境的高级体系结构可能包括但不限于以下组件：

- ◎ 开发人员端点，包括 pc（台式机或笔记本电脑）和虚拟环境，包括基于 pc 和基于云的网络/基础设施设备
- ◎ 服务和应用程序，包括本地的和基于云的
  - o 工具链及其工具（构建工具、打包工具、存储库等）
  - o 漏洞管理（补丁程序和配置）
  - o 版本控制软件和服务
  - o 软件安全审查、分析和测试工具（例如，静态和动态代码分析工具、模糊测试工具、为开发人员提供的即时安全编码培训）
  - o 安全的软件设计工具（例如，威胁建模工具）
- ◎ 构建系统（测试、集成、生产）
- ◎ 配送/交付系统
- ◎ 托管应用程序的生产系统
- ◎ 启用硬件的安全功能来保护私钥

所需的安全功能

本项目旨在使用符合以下特征的商用技术和开源技术开发参考设计和实现：

表 1 中的安全实践适用于整个软件开发生命周期。

只要可行，就使用自动化。

4 相关标准和指导

以下资源和参考资料提供了可以用来帮助开发此解决方案的额外信息：

NIST 框架

- ◎ 改善关键基础设施和网络安全的框架，版本 1.1
- ◎ 风险管理框架(RMF) 概述
- ◎ 安全软件开发框架(SSDF)版本 1.1
- ◎ 网络安全的劳动力框架 （NICE 框架）

NIST 技术项目

- ◎ 信任的硬件根基
- ◎ 国家核对表计划
- ◎ 在线信息性参考资料(OLIR)
- ◎ 开放的安全控制系统评估 语言
- ◎ 安全内容自动化协议 （SCAP）
- ◎ 软件保障参考数据集 （SARD）

NIST 技术指南

- ◎ 应用容器安全指南(SP 800-190)
- ◎ 使用服务网格架构构建基于微服务的安全应用 (SP 800-204A)
- ◎ 系统和组织的网络安全供应链风险管理实践 (SP 800-161 Rev.1)
- ◎ 开发网络弹性系统：一种系统安全工程方法 (SP800-160 第 2 卷修订版。1)
- ◎ 企业补丁管理规划指南：技术预防性维护(SP 800-40 Rev.4)
- ◎ 完全虚拟化技术安全指南(SP 800-125)
- ◎ 硬件支持的安全性：为云计算和边缘计算用例提供分层的平台安全方法 (IR 8320)
- ◎ 用于保护虚拟机(VM)的安全虚拟网络配置(SP 800-125B)
- ◎ 基于服务器的 Hypervisor 平台的安全建议(SP 800-125A Rev.1)
- ◎ 基于微服务的应用系统的安全策略(SP 800-204)
- ◎ 系统安全工程：可信安全系统工程中多学科方法的考虑因素（SP800-160 卷。1)
- ◎ 零信任架构(SP 800-207)

政府、工业、学术界和社区的指导和实践

- ◎ BSA|软件协会
- ◎ 卡耐基梅隆大学(CMU)软件工程研究所(SEI)devsecops 博客
- ◎ 互联网安全中心(CIS) 基准测试
- ◎ 云安全联盟(CSA)DevSecOps 工作 组
- ◎ 信息和软件质量协会(CISQ)标准，实现软件自动化测量
- ◎ 网络安全与信息系统信息分析中心 (CSIAC)
- ◎ 国防信息系统局(DISA)安全技术实施指南 (STIG)
- ◎ 美国国防部 （DoD） 企业级 DevSecOps 倡议
- ◎ 总务管理局(GSA) 关于 DevSecOps 的技术指南
- ◎ 迈克尔·斯科维塔与开源安全联盟合作，开源生态系统中的威胁、风险和缓解措施
- ◎ 微软和 Sogeti，保护企业 DevOps 环境的安全
- ◎ 开源安全基金会(OpenSSF)资源，包括
  - o Alpha-Omega 项目
  - o 开发和发布安全软件的现有准则
  - o 安全工具指南
  - o 开发更安全软件的单页指南
  - o 开源安全指标
  - o OpenSSF 最佳实践奖章计划
  - o 软件包管理器最佳实践

草稿

- 安全审查(开源的软件)
- 安全记分卡-开源的安全健康指标
- sigstore
- SLSA(软件产品的供应链级别）
- 供应链完整性 WG
- 漏洞披露
- WG 确保关键项目的安全

◎ DevOps 和敏捷开发中的安全软件工程国际研讨会的论文和报告

◎ 卓越代码软件保障论坛(SAFECode)关于安全软件开发的出版物， 包括*管理使用第三方组件中固有的安全风险*

5 安全控制流程图

表 1 将美国国家网络安全卓越中心(NCCoE)将应用于这个网络安全挑战的商业和开源产品的特点（如安全软件开发框架（SSDF）实践和任务代表）映射到《改善关键基础设施网络安全框架》、SP800-53，SP800-161 和行政命令(EO)14028 中描述的使用标准和推荐实践。

这些映射表明了执行安全软件开发框架（SSDF）实践和任务如何帮助满足这些其他出版物的要素。本练习旨在展示标准和最佳实践的实际适用性，但并不意味着具有这些特征的产品将满足行业的监管部门审批或认证的要求。

表 1 对已映射的出版物使用以下缩写：

- *EO14028：EO14028，关于改善国家网络安全的行政命令*[6]
- *NISTCSF：NIST 网络安全框架（改善关键基础设施网络安全的框架）*[3]
- *SP80053：SP800-53 第 5 版，针对信息系统和组织的安全和隐私控制*[7]
- *SP800161：SP800-161 修订版 1，针对系统和组织的网络安全供应链风险管理实践*[1]

表 1：安全控制流程图

实践	任务	参考文献
定义软件开发的安全要求 PO. 1)：确保软件开发的安全需求始终是已知的，以便在整个 SDLC 过程中都可以考虑到它们，并	PO.1.1：识别并记录组织的软件开发基础设施和流程的所有安全需求，并随时间维护这些需求。	**EO14028:** 4e (ix) **NISTCSF：** ID.GV-3 **SP80053:** SA-1, SA-8, SA-15, SR-3 **SP800161:** SA-1, SA-8, SA-15, SR-3
	PO.1.2：识别并记录组织开发的软件的所有安全要求，以满足并长期保持这些要求。	**EO14028:** 4e (ix) **NISTCSF：** ID.GV-3 **SP80053:** SA-8, SA-8(3), SA-15, SR-3 **SP800161:** SA-8, SA-15, SR-3
	PO.1.3：将需求告知所有第三方，这些第三方将向组织提供商业软件组件，以供组织自己的软件再利用。[前身为 PW.3.1]	**EO14028:** 4e (vi), 4e (ix) **NISTCSF：** ID.SC-3 **SP80053:** SA-4, SA-9, SA-10, SA-10(1), SA-15, SR-3, SR-4, SR-5 **SP800161:** SA-4, SA-9, SA-9(1), SA-9(3), SA-10, SA-10(1), SA-15, SR-3, SR-4, SR-5
执行角色和职责 PO.2)：确保参与 SDLC 的组织内部和外部的每个人都准备好在整个 SDLC 过程中履行其与 SDLC 相关的角色和职责。	PO.2.1：根据需要创建新的角色，并更改现有角色的职责，以包含 SDLC 的所有部分。定期审查和维护定义的角色和职责，并根据需要更新。	**EO14028:** 4e (ix) **NISTCSF：** ID.AM-6，ID.GV-2 **SP80053:** SA-3 **SP800161:** SA-3
	PO.2.2：为所有有助于安全开发的责任人员提供基于角色的培训。定期审查人员的熟练程度和基于角色的培训，并根据需要更新培训。	**EO14028:** 4e(ix) **NISTCSF:**PR.AT **SP80053:**SA-8 **SP800161：** SA-8
	PO.2.3：获得高级管理人员或授权官方对安全发展的承诺，并将承诺传达给所有与开发相关的职责。	**EO14028:** 4e (ix) **NISTCSF：** ID.RM-1，ID.SC-1
实现支持工具链 PO.3)：使用自动化来减少人力工作，提高整个 SDLC 中安全实践的准确性、可再现性、可用性和全面性，并提供一种方法来记录和演示这些	PO.3.1：指定每个工具链中必须或应该包含哪些工具或工具类型，以减轻已识别的风险，以及如何将工具链组件相互集成。	**EO14028:**4e(iii),4e(ix) **SP80053:** SA-15 **SP800161:** SA-15
	PO.3.2：按照建议的安全实践来部署、运维和维护工具和工具链。	**EO14028:** 4e (i)(F), 4e (ii), 4e (iii), 4e (v), 4e (vi), 4e (ix) **SP80053:** SA-15 **SP800161:** SA-15

实践	任务	参考文献
	PO. 3. 3：配置工具，以生成其支持由组织定义的安全软件开发实践的工件。	**EO14028:** 4e (i)(F), 4e (ii), 4e (v), 4e (ix) **SP80053:** SA-15 **SP800161:** SA-15
定义和使用软件安全检查的标准 PO. 4)：通过定义和使用在开发过程中检查软件的安全性的标准，帮助确保由 SDLC 生成的软件符合组织的期望。	PO. 4. 1：定义整个 SDLC 的软件安全检查和跟踪标准。	**EO14028:** 4e (iv), 4e (v), 4e (ix) **SP80053:** SA-15, SA-15(1) **SP800161:** SA-15, SA-15(1)
	PO. 4. 2：实施过程、机制等。收集和保障支持该标准的必要信息。	**EO14028:** 4e (iv), 4e (v), 4e (ix) **SP80053:** SA-15, SA-15(1), SA-15(11) **SP800161:** SA-15, SA-15(1), SA-15(11)
实现和维护软件开发的安全环境 PO. 5)：确保用于软件开发的环境的所有组件都得到严格保护，免受内部和外部威胁，以防止环境或正在开发或维护的软件收到损害。软件开发环境的示例包括开发、构建、测试和分发环境。	PO. 5. 1：分离和保护涉及到软件开发的每个环境。	**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) **NISTCSF:** PR.AC-5, PR.DS-7 **SP80053:** SA-3(1), SA-8, SA-15 **SP800161:** SA-3, SA-8, SA-15
	PO. 5. 2：保护和加固开发端点（即软件设计人员、开发人员、测试人员、构建人员等的端点）使用基于风险的方法来执行与开发相关的任务。	**EO14028:** 4e (i)(C), 4e (i)(E), 4e (i)(F), 4e (ii), 4e (iii),4e (v), 4e (vi), 4e (ix) **NISTCSF:** PR.AC-4, PR.AC-7, PR.ip-1,PR.IP-3, PR.IP-12,PR.PT-1, PR.PT-3, DE.CM **SP80053:** SA-15 **SP800161:** SA-15
保护所有形式的代码不受未经授权的访问和篡改 PS. 1)：帮助防止对代码进行未经授权的更改，包括无意的和有意的，这可能会规避或否定软件的预期安全特性。对于不打算公开访问的代码，这有助于防止软件被盗，并可能使攻击者发现软件中的漏洞更困难或更耗时。	PS. 1. 1：基于最小特权的原则存储所有形式的代码，包括源代码、可执行代码和作为代码的配置，以便只有经授权的人员、工具、服务等，有访问权限。	**EO14028:** 4e (iii), 4e (iv), 4e (ix) **NISTCSF:** PR.AC-4, PR.DS-6, PR.IP-3 **SP80053：** SA-10 **SP800161:** SA-8, SA-10
提供一种验证软件发布完整性的机制 PS. 2)：帮助软件获取者确保他们所获取的软件是合法的，并且没有被篡改。	PS. 2. 1：向软件获取者提供软件完整性验证信息。	**EO14028:** 4e (iii), 4e (ix), 4e(x) **NISTCSF:** PR.DS-6 **SP80053:** SA-8 **SP800161:** SA-8

实践	任务	参考文献
存档和保护每一个软件版本 PS.3)： 保留软件版本，以帮助识别、分析和消除发布后在软件中发现的漏洞。	**PS.3.1：** 安全地归档每个软件版本要保留的必要文件和支持数据（例如，完整性验证信息、来源数据）。	**EO14028:** 4e (iii), 4e (vi), 4e (ix), 4e (x) **NISTCSF:** PR.IP-4 **SP80053:** SA-10, SA-15, SA-15(11), SR-4 **SP800161:** SA-8, SA-10, SA-15(11), SR-4
	PS.3.2： 收集、保护、维护和共享每个软件发布的所有组件的来源数据(例如，在一个软件材料清单[SBOM]中)。	**EO14028:** 4e (vi), 4e (vii), 4e (ix), 4e (x) **SP80053:** SA-8, SR-3, SR-4 **SP800161:** SA-8, SR-3, SR-4
设计一种能够满足安全要求并降低安全风险的软件 PW.1)： 识别并评估软件的安全要求；确定软件在运行过程中可能面临的安全风险，以及软件的设计和架构应如何减轻这些风险；并对基于风险分析表明应放宽或放弃安全要求的任何情况进行论证。在软件设计过程中解决安全需求和风险（按设计来实现安全）是提高软件安全性的关键，也有助于提高开发效率。	**PW.1.1：** 使用风险建模的形式，如威胁建模、攻击建模或攻击面映射 -帮助评估该软件的安全风险。	**EO14028:** 4e (ix) **NISTCSF：** ID.RA **SP80053:** SA-8, SA-11(2), SA-11(6), SA-15(5) **SP800161:** SA-8, SA-11(2), SA-11(6), SA-15(5)
	PW.1.2： 跟踪和维护软件的安全要求、风险和设计决策。	**EO14028:** 4e (v), 4e (ix) **SP80053:** SA-8, SA-10, SA-17 **SP800161:** SA-8, SA-17
	PW.1.3： 在适当的情况下，构建对使用标准化安全特性和服务的支持（例如，使软件能够与现有的日志管理、身份管理、访问控制和漏洞管理系统集成），而不是创建安全特性和服务的专有实现。[原标题为 PW.4.3]	**EO14028:** 4e (ix)
审查软件设计，以验证是否符合安全要求和风险信息 PW. 2)： 帮助确保该软件将满足安全要求，并令人满意地处理已识别的风险信息。	**PW.2.1：** 让 1)未参与设计的合格人员（或人员）2)工具链中实例化的自动化流程审查软件设计，以确认和执行其满足所有安全要求，并满意地处理识别的风险信息。	**EO14028:** 4e (iv), 4e (v), 4e (ix)
验证第三方软件符合安全要求 PW.3)： *移到 PW.4*	***PW.3.1：*** *移至 PO.1.3*	
	PW.3.2： *移至 PW.4.4*	
在可行的情况下，重新使用现有的、安全性能好的软件，而不是复制功能 PW. 4)： 通过重用已检查其安全状态的软件模块和服务，降低软件开发成本，加快软件开发，并降低向软件中引入额外安全漏洞的可能性。这对于实现安全功能的软件尤其重要，例如加密模块和协议。	**PW.4.1：** 从商业软件、开源软件和其他第三方开发人员那里获取并维护安全良好的软件组件（例如，软件库、模块、中间件、框架），以供组织的软件使用。	**EO14028:** 4e (iii), 4e (vi), 4e (ix), 4e (x) **NISTCSF：** ID.SC-2 **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

实践	任务	参考文献
	PW.4.2： 根据 SDLC 流程创建和维护安全良好的内部软件组件，以满足第三方软件组件不能更好地满足的常见内部软件开发需求。	**EO14028:** 4e (ix) **SP80053:** SA-8(3) **SP800161:** SA-8(3)
	PW.4.3： *移至 PW.1.3*	
	PW.4.4： 验证所获得的商业、开源和所有其他第三方软件组件在其整个生命周期中都符合本组织所定义的要求。	**EO14028:** 4e (iii), 4e (iv), 4e (vi), 4e (ix), 4e (x) **NISTCSF：** ID.SC-4, PR.DS-6 **SP80053:** SA-9, SR-3, SR-4, SR-4(3), SR-4(4) **SP800161:** SA-4, SA-8, SA-9, SA-9(3), SR-3, SR-4, SR-4(3), SR-4(4)
	PW.4.5： *移到 PW.4.1 和 PW.4.4*	
通过坚持安全编码实践来创建源代码 PW.5)： 减少软件中的安全漏洞的数量，并通过最小化在源代码创建过程中引入的符合或超过组织定义的漏洞严重性标准的漏洞来降低成本。	**PW.5.1：** 遵循所有适合于开发语言和环境的安全编码实践，以满足组织的要求。	**EO14028:** 4e (iv), 4e (ix)
	PW.5.2： *以 PW.5.1 为例*	
配置编译、解释器和构建进程，以提高可执行文件的安全性 PW.6)： 通过在测试发生前消除安全漏洞，减少软件中的安全漏洞数量并降低成本。	**PW.6.1：** 使用编译器、解释器和构建工具，提供改善可执行安全性的特性。	**EO14028:** 4e (iv), 4e (ix) **SP80053:** SA-15 **SP800161:** SA-15
	PW.6.2： 确定应该使用哪些编译器、解释器和构建工具的功能，以及每个功能应该如何配置，然后实现和使用已批准的配置。	**EO14028:** 4e (iv), 4e (ix) **SP80053:** SA-15, SR-9 **SP800161:** SA-15, SR-9
审查/分析人类可读的代码，以识别漏洞，并验证是否符合安全要求 PW.7)： 帮助识别漏洞，以便在软件发布之前纠正这些漏洞，以防止漏洞被利用。使用自动化方法可以降低检测漏洞所需的工作量和资源。人类可读代码包括源代码、脚本和组织认为人类可读的任何其他形式的代码。	**PW.7.1：** 根据组织规定，确定是否应该使用代码审查（人员直接查看代码以发现问题）/代码分析（工具用于发现代码中的问题，可以是完全自动化的方式，也可以是与一个人一起使用）。	**EO14028:** 4e (iv), 4e (ix) **SP80053:** SA-11 **SP800161:** SA-11
	PW.7.2： 根据组织的安全编码标准进行代码审查/代码分析，并在开发团队的工作流程或问题跟踪系统中记录和处理所有发现的问题和建议的补救措施。	**EO14028:** 4e (iv), 4e (v), 4e (ix) **SP80053:** SA-11, SA-11(1), SA-11(4), SA-15(7) **SP800161:** SA-11, SA-11(1), SA-11(4), SA-15(7)

实践	任务	参考文献
测试可执行代码，以识别漏洞并验证是否符合安全要求 PW.8)： 帮助识别漏洞，以便在软件发布之前纠正这些漏洞，以防止漏洞被利用。使用自动化的方法降低了检测漏洞所需的工作和资源，并提高了可跟踪性和可重复性。可执行代码包括二进制文件、直接执行的字节码和源代码，以及组织认为可执行的任何其他形式的代码。	**PW.8.1：** 确定是否应该执行可执行代码测试，以发现先前的审查、分析或测试没有识别出的漏洞，如果是，应该使用哪种类型的测试。	**EO14028:** 4e (ix) **SP80053:** SA-11 **SP800161:** SA-11
	PW.8.2： 确定测试范围、设计测试、执行测试并记录结果，包括记录和试用开发团队的工作流程或问题跟踪系统中所有发现的问题和建议的补救措施。	**EO14028:** 4e (iv), 4e (v), 4e (ix) **SP80053:** SA-11, SA-11(5), SA-11(8), SA-15(7) **SP800161:** SA-11, SA-11(5), SA-11(8), SA-15(7)
配置软件使其具有默认的安全设置 PW.9)： 帮助提高软件在安装时的安全性，以降低软件在弱安全设置下部署的可能性，使其面临更大的破坏风险。	**PW.9.1：** 通过确定如何配置对安全或安全相关设置有影响的每个设置来定义安全基线，以便默认设置是安全的，并且不会削弱平台、网络基础设施或服务提供的安全功能。	**EO14028:** 4e (iv), 4e (ix)
	PW.9.2： 实现默认设置（或默认设置组，如果适用），并为软件管理员记录每个设置。	**EO14028:** 4e (iv), 4e (ix) **SP80053:** SA-5, SA-8(23) **SP800161:** SA-5, SA-8(23)
持续地识别和确认漏洞 RV.1)： 帮助确保更快地识别漏洞，以便根据风险更快地修复它们，减少攻击者的机会窗口。	**RV.1.1：** 从软件获取者、用户和公共资源中收集有关软件和第三方组件的潜在漏洞的信息，并调查所有可信的报告。	**EO14028：** 4e(iv)、4e(vi)、4e(viii)、4e(ix) **SP80053：** SA-10、SR-3、SR-4 **SP800161：** SA-10、SR-3、SR-4
	RV.1.2： 检查、分析/测试软件的代码，以识别或确认是否存在以前未检测到的漏洞。	**EO14028：** 4e(iv)、4e(vi)、4e(viii)、4e(ix) **SP80053:** SA-11 **SP800161:** SA-11
	RV.1.3： 制定一个解决漏洞披露和补救问题的策略，并实施支持该策略所需的角色、职责和流程。	**EO14028：** 4e(viii)，4e(ix) **SP80053：** SA-15（10） **SP800161：** SA-15（10）
评估、确定优先级和补救漏洞 RV.2)： 帮助确保根据风险修复漏洞，以减少攻击者的机会窗口。	**RV.2.1：** 分析每个漏洞，以收集足够的风险信息，以计划其补救或其他风险响应。	**EO14028：** 4e(iv），4e(viii)，4e(ix) **SP80053：** SA-10、SA-15（7） **SP800161：** SA-15（7）

实践	任务	参考文献
	RV.2.2： 规划并实施针对漏洞的风险响应。	**EO14028：** 4e(iv)、4e(vi)、4e(viii)、4e(ix) **SP80053:** SA-5, SA-10, SA-11, SA-15(7) **SP800161:** SA-5, SA-8, SA-10, SA-11, SA-15(7)
分析漏洞，以确定其根本原因 RV.3)： 帮助减少未来漏洞出现的频率。	**RV.3.1：** 分析已识别的漏洞，以确定其根本原因。	**EO14028:** 4e (ix)
	RV.3.2： 随着时间的推移，分析根本原因，以识别模式，例如没有一致遵循某个特定的安全编码实践。	**EO14028:** 4e (ix)
	RV.3.3： 检查软件中类似的漏洞，以消除一类漏洞，并主动修复它们，而不是等待外部报告。	**EO14028：** 4e(iv)、4e(viii)、4e(ix) **SP80053:** SA-11 **SP800161:** SA-11
	RV.3.4： 检查 SDLC 过程，并在适当的情况下对其进行更新，以防止（或减少）在软件更新或已创建的新软件更新中反复出现的根本原因。	**EO14028:** 4e (ix) **SP80053:** SA-15 **SP800161:** SA-15

附录 A 参考文献

[1] J.Boyens 等人，《系统和组织的网络安全供应链风险管理实践》，美国国家标准与技术研究所(NIST)特别出版物(SP)800-161 修订版，盖瑟斯堡，马里兰州，2022 年 5 月，326 页。可用：https://doi.org/10.6028/NIST.SP.800-161r1

[2] M.Souppaya 等人，安全软件开发框架(SSDF)第 1.1 版：《减轻软件漏洞风险的建议》，美国国家标准与技术研究所(NIST)特别出版物(SP)800-218，盖瑟斯堡，马里兰州，2022 年 2 月，第 36 页。可用：https://doi.org/10.6028/NIST.SP.800-218

[3] NIST，改善关键基础设施网络安全的框架，第 1.1 版，2018 年。https://doi.org/10.6028/NIST.CSWP.04162018

[4] M.Souppaya 等人，应用容器安全指南，美国国家标准与技术研究所(NIST)特别出版物(SP)800-190，盖瑟斯堡，马里兰州，2017 年 9 月，63 页。可用：https://doi.org/10.6028/NIST.SP.800-190

[5] E. LeMay 等人，常见的滥用评分系统(CMSS)：软件的指标 特征滥用漏洞，美国国家标准与技术研究所(NIST) 内部报告(IR)7864，盖瑟斯堡，马里兰州，2012 年 7 月，第 39 页。可用： https://doi.org/10.6028/NIST.IR.7864

[6] 关于改善国家网络安全的行政命令，行政命令(EO)14028,2021 年 5 月 12 日。可用性：https://www.whitehouse.gov/briefing-room/presIDential-actions/2021/05/12/executive-order-on-improving-the-nations-cybersecurity/

[7] 信息系统和组织安全与隐私控制，美国国家标准与技术研究所(NIST)特别出版物(SP)800-53 修订 5，盖瑟斯堡，马里兰州，2020 年 9 月，492 页。可用：https://doi.org/10.6028/NIST.SP.800-53r5

附录 B：首字母缩略词和缩写

ATO	运营授权
CI/CD	持续集成/持续交付
CIS	互联网安全中心
CISQ	信息和软件质量联盟
CMU	卡内基·梅隆大学
CSA	云安全联盟
C-SCRM	网络安全与供应链风险管理
CSIAC	网络安全与信息系统信息分析中心
DevOps	软件开发和 IT 运维
DevSecOps	软件开发、安全和 IT 运维
DISA	国防信息系统局
DoD	国防部
EO	行政命令
FOSS	免费的和开源的软件
GSA	总务管理局
IoT	物联网
IT	信息技术
NCCoE	美国国家网络安全卓越中心
NICE	美国国家网络安全教育倡议
NIST	美国国家标准与技术研究所
OLIR	在线信息参考
OpenSSF	开源安全基金会
OT	运维技术
PC	个人电脑
RMF	风险管理框架
SaaS	软件即服务
SAFECode	卓越代码软件保障论坛
SARD	软件保证参考数据集
SBOM	软件材料清单
SCAP	安全内容自动化协议
SDLC	软件开发生命周期

SEI	软件工程研究所
SLSA	软件工件的供应链级别
SP	特别出版物
SSDF	安全软件开发框架
STIG	安全技术实施指南
VM	虚拟机