ICS 35.240.99

ZMDS

中关村医疗器械产业技术创新联盟团体标准

T/ZMDS 20008—2024

医疗器械嵌入式软件漏洞评估方法

Method for vulnerability assessment of embedded software in medical devices

2024-12-30 发布　　2024-12-30 实施

中关村医疗器械产业技术创新联盟　　发布

目 次

前　言

本文件的某些内容可能涉及专利。本文件的发布机构不承担识别相关专利的责任。

本文件由北京水木金昇科技服务有限公司提出并归口。

本文件起草单位：北京水木金昇医疗科技服务有限公司、北京中关村水木医疗科技有限公司、上海安般信息科技有限公司、北京智精灵科技有限公司、中国卫生信息与健康医疗大数据学会信息及应用安全防护分会、创领心律管理医疗器械（上海）有限公司、哈尔滨工业大学（威海）、北京科技大学、中关村华安关键信息基础设施安全保护联盟。

本文件主要起草人：冯健、王斌、王春燕、王骁、王晓怡、张坤、张建林、张建锋、王凯、曹伟娜、徐源。

引 言

随着医疗技术的快速发展，医疗器械日益依赖于复杂的数据交互功能，这使得其运行的网络环境异常复杂。在提升医疗服务效率和质量的同时，患者对安全和健康的高标准要求，使得嵌入式软件的广泛应用必须不断提升可靠性和安全性，网络安全漏洞的存在使医疗器械面临可能被恶意攻击的风险，这不仅可能导致设备功能故障，还可能引发医疗敏感数据的泄露。

在此背景下，对于嵌入式软件的漏洞评估方法尤其重要。本文件从医疗器械嵌入式软件整个生命周期的各个部分，阐述了漏洞评估的重要性，在验证确认阶段使用的不同评估方法（如：对于未包含现成软件(组件)以及采用私有协议进行数据交换的嵌入式系统，可通过代码审计、固件包/二进制文件扫描、攻击面分析、漏洞链分析等方式进行验证）。

本标准旨在提供一种针对医疗器械嵌入式软件的漏洞评估方法，推进实施系统的漏洞管理流程不仅有助于减小医疗器械面临安全风险时的影响，还能通过定期的进行漏洞评估和维护，有效降低医疗器械网络安全风险。

医疗器械嵌入式软件漏洞评估方法

1 范围

本标准规定了针对具有数据交换接口的嵌入式医疗器械软件的漏洞评估方法。无论设备是完全联网还是仅具有限的联网权限，文中数据交换接口均包含调试接口。本标准旨在为医疗设备制造商、软件开发者及相关监管机构提供具体的操作指南，确保医疗器械网络安全的系统性和全面性。

2 规范性引用文件

下列文件对于本文的应用参考是必不可少的。凡是注日期的引用文件，仅所注日期的版本 适用于本文件。凡是不注日期的引用文件，其最新版本（包括所有的修改单）适用于本文件。

GB/T 28458-2020 信息安全技术 网络安全漏洞标识与描述规范

YY/T 0664-2020 医疗器械软件 软件生命周期过程

GB／T 42062-2022 医疗器械 风险管理对医疗器械的应用

GB/T 20984-2022 信息安全技术信息安全风险评估规范

GB/T 25069-2022 信息安全技术 术语

3 术语和定义、缩略语

3.1 术语和定义

下列术语和定义适用于本文件。

3.1.1

端口 port

连接的端点。

注：在互联网协议的语境下，端口是传输控制协议（TCP）连接或用户数据报协议（UDP） 消息的逻辑信道端点。基于 TCP 或 UDP 的应用协议，通常已分配默认端口号，如为超文本传输协议（HTTP）的端口号是 80。

［来源：GB/T 25069-2022,3.130］

3.1.2

访问控制 access control

一种确保数据处理系统的资源只能由经授权实体以授权方式进行访问的手段。

［来源：GB/T 25069-2022,3.147］

3.1.3

固件 firmware

功能上独立于主存储器，通常存储在只读存储器（ROM）中的指令和相关数据的有序集。

[来源：GB/T 25069-2022,3.225]

3.1.4

过滤 filtering

依照所规定的准则，接受或拒绝数据流通过某一网络的过程。

[来源：GB/T 25069-2022,3.235]

3.1.5

基于角色的访问控制 role-based access control

一种对某一角色授权，许可其访问相应对象的访问控制方法。

[来源：GB/T 25069-2022,3.263]

3.1.6

威胁 threat

可能对系统或组织造成危害的不期望事件的潜在因素。

[来源：GB/T 25069-2022,3.628]

3.1.7

主机 host

在基于传输控制协议/互联网协议(TCP/IP)的网络(如互联网)中，可设定地址的系统或计算机。

[来源：GB/T 25069-2022,3.803]

3.2 缩略语

下列缩略语适用于本文件。

CVSS：通用漏洞评分系统(Common Vulnerability Scoring System)

CVE：通用漏洞披露(Common Vulnerabilities and Exposure)

CNVD：中国国家信息安全漏洞共享平台(China National Vulnerability Database)

CNNVD： 中国国家信息安全漏洞库(China National Vulnerability Database of Information Security)

SBOM：软件物料清单(Software Bill of Materials)

NVD：国家漏洞数据库 (National Vulnerability Database)

OTS：现成的解决方案 (Off-the-Shelf Solution)

OWASP：开放式Web应用安全项目 (Open Web Application Security Project)

CERT：计算机应急响应小组 (Computer Emergency Response Team)

TCP：传输控制协议 (Transmission Control Protocol)

UDP：用户数据报协议 (User Datagram Protocol)

BT：蓝牙 (Bluetooth)

BLE：低功耗蓝牙 (Bluetooth Low Energy)

WLAN：无线局域网络(Wireless Local Area Network)

USB：通用串行总线 (Universal Serial Bus)

CAN：控制器局域网络 (Controller Area Network)

STRIDE：微软提出的一种威胁建模方法，威胁类型分为Spoofing(仿冒)、Tampering(篡改)、Repudiation(抵赖)、Information Disclosure(信息泄漏)、Denial of Service(拒绝服务)和Elevation of Privilege(权限提升)六种威胁构成

4 软件生命周期

4.1 软件生命周期概述

医疗器械嵌入式软件的开发和维护需要遵循严格的软件生命周期模型，以确保产品的安全性、有效性和合规性。本章将介绍适用于医疗器械软件的生命周期模型，以及在各个阶段需要开展的安全活动，特别强调漏洞评估的重要性。

4.2 软件生命周期模型

软件生命周期模型是指导软件从需求分析到维护退役全过程的框架。对于医疗器械软件，常用的生命周期模型包括：

1. 瀑布模型（Waterfall Model）：这是最传统的软件开发模型，按照顺序执行需求分析、设计、实现、测试、部署和维护各个阶段。每个阶段的输出作为下一个阶段的输入，阶段之间严格区分，适用于需求明确且变化较少的项目，（瀑布模型示意图见附录A）。

2. V模型（V-Model）：V模型是瀑布模型的扩展，强调验证和确认（Verification and Validation）。开发过程的左侧是逐级细化的开发活动，右侧是对应的测试和验证活动。V模型适用于对安全性和可靠性要求极高的系统，如医疗器械软件，有助于确保每个开发阶段的产出都经过严格的验证，（V模型示意图见附录B）。

3. 敏捷开发模型（Agile Development Model）：敏捷开发强调迭代和增量式的开发方法，通过小规模、快速的迭代应对需求变化。常见的敏捷方法包括Scrum、Kanban等，适用于需求变化频繁、需要快速响应的项目。然而，在医疗器械软件开发中，需要谨慎应用敏捷方法，确保满足监管机构的合规要求。

医疗器械软件生命周期模型的选择应基于产品的复杂性、风险等级和监管要求。通常情况下，V模型被广泛应用于医疗器械软件开发，因为它提供了明确的验证和确认路径，确保每个阶段的输出都符合预期。此外，结合敏捷开发的实践，可以提高开发效率和产品质量，但必须确保过程的可追溯性和合规性。

5 设计阶段

5.1 安全评估任务

医疗器械制造商应在设计初期明确产品的安全评估任务。这些任务应基于对患者安全、数据完整性和系统可用性的考虑，符合国际标准和法规要求。安全评估任务包括：

- 明确关键资产：确定需要保护的业务系统、患者数据和其他敏感信息。
- 定义安全需求：基于常见威胁分析，制定具体的安全测试需求。
- 确定安全目标：设定可测量的安全性能指标，以评估安全目标的实现程度。

5.2 网络安全需求

网络安全需求是确保嵌入式软件在整个生命周期内保持安全性的基础。这些需求应基于设计阶段制定的安全目标：

- 功能性安全需求：确保软件在各种操作条件下的可靠性和安全性。例如，防止未经授权的访问、确保数据的完整性和可用性。
- 自身安全需求：涵盖认证、授权、数据加密、日志记录和审计等方面，以保障系统的整体安全性。同时，应考虑对关键安全机制在复杂环境（网络异常、系统故障、恶意攻击）下的防护能力要求，并在需求中明确相应的安全指标和容忍度。

5.3 SBOM清单

软件物料清单（SBOM）是对嵌入式软件所包含的所有组件、库和依赖项的全面清单。根据国际最佳实践，SBOM应包含以下信息：

- 组件名称和版本：明确标识每个软件组件及其版本；同时，建议在此处注明Off-the-Shelf（OTS）软件的服务支持终止时间（End-of-Support / End-of-Life），以便及时评估并应对潜在安全风险。
- 供应商信息：提供组件的原始开发者或供应商信息。
- 许可证和版权信息：列出组件的使用许可和版权声明，确保法律合规性。
- 组件关系和依赖性：描述组件之间的关系，识别潜在的供应链风险。
- 已知漏洞信息：关联组件与已公开的漏洞数据库（如CVE），便于后续的漏洞管理。

维护详尽的SBOM有助于：

- 漏洞管理：快速识别和响应已知的安全漏洞。
- 合规性审查：满足法规和标准对软件透明度的要求。
- 供应链风险控制：监控第三方组件的安全性，降低供应链攻击的风险。

通过在设计阶段积极制定安全评估任务、明确网络安全需求和建立SBOM清单，医疗器械制造商可以显著提高嵌入式软件的安全性，为后续的研发和验证阶段奠定坚实的基础。

6 医疗器械研发阶段

6.1 威胁建模

威胁建模是识别、分析和评估系统潜在威胁的系统性方法。根据国际标准（如IEC 62443和ISO 14971），威胁建模应包括以下步骤：

- 系统分解：详细描述系统架构、数据流和组件交互。
- 识别潜在威胁：利用方法论（如STRIDE模型）识别可能的威胁类型，包括仿冒、篡改、否认、信息泄露、拒绝服务和权限提升。
- 评估风险：根据威胁的可能性和影响程度，对风险进行定性或定量评估。
- 制定缓解措施：针对高风险威胁，设计并实施适当的安全控制措施。

6.2 已知漏洞评估

已知漏洞评估是识别和分析软件中已知安全漏洞的过程，旨在降低系统遭受攻击的风险。此过程应结合国际最佳实践和标准，确保漏洞评估的全面性和有效性。评估步骤参考本文【章节10 风险评估】。

6.3 网络安全架构设计

网络安全架构设计是构建嵌入式软件系统时，确保其具备抵御各种网络攻击能力的关键步骤。一个健全的安全架构不仅能防范已知威胁，还应具备应对未知攻击类型的弹性。架构设计原则如下：

- 最小权限原则：每个组件和用户仅具备完成其功能所需的最低权限，减少潜在的攻击面。
- 纵深防御：采用多层次的安全防护措施，确保即使一层防御被突破，其他层次仍能保护重要业务系统和数据，同时，可触发应急响应机制，增强整体安全性，提高系统的抗风险能力。
- 业务驱动：基于业务风险驱动设计安全架构，寻找业务创造价值和抵御风险成本的平衡点，安全架构既要可抵御外部风险，又要可支撑业务需求。

7 验证确认阶段(漏洞评估)

7.1 已知漏洞评估

对于医疗器械嵌入式软件漏洞的评估，不仅针对医疗器械产品本身，还应综合考虑产品实际使用时所处的环境。产品技术要求中所描述的必备软硬件、运行环境也应在评估的范围之内。在进行医疗器械嵌入式软件漏洞评估时，注册申请人应根据产品技术要求的内容，确保产品运行所必需的其他医疗器械软件、医用中间件及医疗器械硬件产品处于正常的配置和运行条件下。

同时在进行嵌入式软件漏洞评估时，应确保医疗器械运行在产品技术要求所规定的典型运行环境中，包括硬件配置、外部软件环境、必备软件、网络条件等。

(1) 环境配置确认

- 确认嵌入式软件运行所需的硬件配置和外部软件环境符合产品技术要求。
- 确保所有必备软件和中间件处于推荐的版本和配置状态。

(2) 组件识别与清单核对

- 利用SBOM识别所有第三方组件、库及其版本。
- 核对实际运行环境中的组件版本，确保与SBOM一致。

(3) 漏洞匹配与识别

- 将SBOM中的组件与已知漏洞数据库（如NVD、CVE、CNVD、CNNVD）进行匹配，识别存在的已知漏洞。
- 评估每个漏洞的严重性和影响，依据CVSS评分进行分类。

(4) 风险评估与优先级排序

- 根据漏洞的严重性、易利用性及对系统的潜在影响，评估其风险等级。
- 对高风险漏洞进行优先处理，制定修复或缓解措施。

(5) 缓解措施实施

- 对于高风险漏洞，及时更新组件版本、应用补丁或增强安全控制措施。
- 记录所有缓解措施的实施过程，确保可追溯性。

7.2 漏洞扫描

漏洞扫描是通过自动化工具检测嵌入式软件中的已知安全漏洞和潜在弱点的过程。此过程应系统化、全面化，确保所有可能的漏洞点都得到有效识别和评估。

7.2.1漏洞扫描步骤

(1) 扫描范围定义

- 明确目标范围，包括嵌入式软件的各个模块、组件、OTS软件、运行环境及其依赖项。
- 确定扫描频率和时间安排，确保定期进行全面扫描。

(2) 扫描配置与执行

- 根据扫描工具的要求，配置扫描参数和规则集。
- 执行漏洞扫描，确保覆盖所有关键组件和功能模块。

(3) 扫描结果分析

- 对扫描生成的漏洞报告进行详细分析，确认漏洞的真实性和严重性。
- 过滤误报和低优先级漏洞，聚焦于高危和关键漏洞。

(4) 漏洞修复与验证

- 针对确认的漏洞，制定并实施修复计划，包括代码修复、配置更改或补丁应用。
- 重新扫描修复后的软件，验证漏洞是否已被有效解决。

7.2.2 扫描频率与持续监控

- 定期扫描：建议每月或每个开发迭代周期进行一次全面漏洞扫描，确保及时发现和修复新出现的漏洞。
- 持续监控：利用持续集成工具，实时监控代码库的安全状态，自动触发漏洞扫描，及时响应安全威胁。

7.2.3 扫描报告与文档化

- 扫描报告：生成详细的漏洞扫描报告，包含发现的漏洞类型、严重性等级、影响范围及修复建议。
- 文档记录：保存所有扫描报告和修复记录，确保漏洞管理过程的透明性和可追溯性。

7.3 代码审计

代码审计是通过手动或自动化手段对嵌入式软件源代码进行详细检查，以识别潜在的安全漏洞和编码缺陷。代码审计有助于发现那些难以通过自动化工具检测到的复杂漏洞，提升软件的整体安全性。

7.3.1 代码审计方法

静态代码分析：使用静态代码分析工具自动扫描源代码，识别常见的安全漏洞和编码错误。

手动代码审计：由经验丰富的安全专家或开发人员手动审查代码，重点检查关键模块和敏感功能，发现复杂和隐蔽的漏洞。

7.3.2 代码审计步骤

(1) 审计计划制定
- 确定审计的目标范围和重点，包括关键模块、敏感功能和高风险区域。
- 制定详细的审计计划，明确审计的时间安排、责任人和方法。

(2) 工具配置与预审
- 配置静态代码分析工具，选择适当的规则集和扫描参数。
- 进行预审，识别明显的低风险漏洞，集中精力处理高风险问题。

(3) 详细审计与分析
- 手动审计代码，检查潜在的逻辑漏洞、输入验证缺陷、资源管理问题等。
- 分析代码中的安全控制措施，确保其有效性和正确性。
- 分析使用的现成软件是否存在已知漏洞。

(4) 漏洞修复与优化
- 针对发现的漏洞，制定修复方案，修改代码或优化安全控制。
- 进行修复后的代码复审，确保漏洞已被有效解决且未引入新的问题。

7.3.3 审计标准与指南

- OWASP安全编码规范：遵循OWASP提供的安全编码最佳实践，避免常见的安全漏洞。
- CERT编码标准：参考CERT发布的编码标准和指南，提升代码的安全性和可靠性。
- 公司内部安全标准：遵循组织内部制定的代码审计标准和流程，确保一致性和全面性。

7.3.4 审计报告与文档化

- 审计报告：记录审计过程中发现的所有漏洞和缺陷，详细描述其性质、影响和修复建议。
- 修复记录：保存所有漏洞修复的详细记录，确保修复过程的透明性和可追溯性。
- 审计总结：总结审计结果和改进建议，推动整体代码质量和安全性的持续提升。

7.4 固件包/二进制文件扫描

基于不依赖任何数据交互接口的嵌入式软件，可以通过静态分析程序特征来发现漏洞，而无需执行程序代码。由于固件程序通常涉及知识产权，源代码很少公开，因此需要对固件文件进行逆向工程，并利用静态分析技术进行深入分析。在能够运行的动态沙箱环境下，也可以进行动态扫描以检测潜在漏洞。

7.4.1 扫描步骤

(1) 固件提取与解析
- 使用固件分析工具提取固件包中的文件系统、配置文件和可执行文件。
- 解析固件结构，识别各个组件和依赖关系。

(2) 静态扫描与漏洞识别
- 对提取的二进制文件进行静态扫描，识别已知的漏洞模式和安全缺陷。
- 分析二进制文件中的安全控制措施，确保其有效性。

(3) 动态分析与行为检测
- 在受控环境中运行二进制文件，监控其行为，检测异常活动和潜在的安全威胁。
- 利用符号执行工具，模拟不同输入和操作条件，发现潜在的逻辑漏洞和安全缺陷。

(4) 漏洞评估与修复
- 对发现的漏洞进行风险评估，确定其严重性和优先级。
- 制定并实施修复计划，更新固件包或二进制文件，修复已识别的漏洞。

7.4.2 扫描频率与更新

- 发布前扫描：在嵌入式软件发布前，进行全面的固件包和二进制文件扫描，确保产品的安全性。
- 持续监控：在产品发布后，定期扫描固件包和二进制文件，及时发现和修复新出现的漏洞。

7.4.1 扫描报告与文档化

- 扫描报告：生成详细的扫描报告，记录发现的所有漏洞、其位置、严重性等级及修复建议。
- 修复验证：对修复后的固件包和二进制文件重新进行扫描，验证漏洞是否已被有效解决。
- 文档记录：保存所有扫描报告和修复记录，确保漏洞管理过程的透明性和可追溯性。

7.5 硬件接口模糊测试

硬件接口的模糊测试（Fuzzing）是一种有效的技术，用于识别嵌入式软件中可能未被常规测试流程发现的安全漏洞。该方法适用于多种接口类型，包括但不限于BT、BLE、RJ45、WLAN、RS232、RS485、USB、CAN（控制器局域网络）等。通过定制化的模糊测试方法和策略，针对性地发现和评估潜在的安全漏洞，确保医疗器械在面对异常或恶意输入时的稳健性和安全性。

7.5.1 流程

模糊测试通常分为四个阶段：接口识别、功能评估、策略定制、分析评估与报告。以下详细描述各阶段的实施步骤和关键要点。

第一阶段-接口识别：

确定所检测硬件接口的种类和特点。

列出设备所有支持的硬件接口类型，如BLE、RS232、RS485、USB、CAN等

确认每种接口的物理和逻辑特性，包括通信协议、数据传输速率、信号类型等

注：若没有协议信息，使用脏数据测试

第二阶段-功能评估：

评估每种接口在设备中的作用和重要性，例如数据传输、设备控制或诊断功能。

根据接口在设备操作中的关键性，评估其安全性需求和优先级（识别对患者安全和数据完整性影响较大的接口功能）。

确认接口与其他系统组件或外部设备的依赖关系，评估这些依赖关系对接口安全性的潜在影响。

第三阶段-策略定制：

根据已识别的接口特性和功能需求，设计针对性的模糊测试用例和策略。

基于接口规范和功能需求，设计覆盖各种异常和极端情况的测试用例，生成无效、异常或随机的数据包，模拟不同的攻击场景和错误输入。

确定模糊测试的深度和广度，包括输入数据的复杂性和变异方式，定义测试的执行频率和持续时间，确保全面覆盖。

第四阶段-分析、评估与报告：

分析模糊测试的结果，识别异常响应和系统崩溃，评估漏洞的严重性并生成详细报告。

根据漏洞的性质和影响，分类如缓冲区溢出、异常处理错误、权限提升等，采用CVSS（通用漏洞评分系统）评分标准，对漏洞进行严重性评级，确定修复优先级。评估步骤参考本文【章节10 风险评估】

模糊测试报告应包含测试范围、方法、发现的漏洞及其详细描述。

7.6 攻击面分析

攻击面分析是识别和评估嵌入式软件系统中所有潜在的攻击入口点和弱点的过程。

7.6.1 流程

第一阶段-接口识别：

系统组件分析：详细分析系统架构，识别所有软件组件、硬件接口和通信渠道。

数据流分析：研究系统中的数据流动路径，识别敏感数据的存储、传输和处理环节（敏感数据典型实例见附录C）。

用户交互点识别：确定系统与用户或其他设备交互的所有接口和入口点。

第二阶段-攻击面评估：

入口点风险评估：评估每个入口点的潜在风险，包括未经授权的访问、数据泄露和系统操控等。

漏洞分布分析：分析已识别漏洞在系统中的分布情况，确定高风险区域和关键防护点。

威胁模型结合：将攻击面分析与威胁模型相结合，识别特定威胁情景下的攻击路径和风险点。

第三阶段-防护策略制定：

减少攻击面：通过简化系统架构、移除不必要的功能和接口，减少潜在的攻击入口点。

强化防护措施：在关键入口点部署（或实施）防护措施，如访问控制、加密通信。

持续监控与响应：建立持续的监控机制，实时检测和响应攻击行为，及时修复发现的安全缺陷。

第四阶段-攻击面分析报告

分析报告：详细记录攻击面识别和评估的过程、发现的潜在风险及其影响，提供相应建议。

改进措施记录：保存所有防护策略的制定和实施过程，确保系统的安全性持续提升。

定期更新：随着系统的演进和新威胁的出现，定期更新攻击面分析，确保安全策略的有效性。

7.7 漏洞链分析

漏洞链分析是研究多个漏洞之间的关联和组合，评估它们在攻击过程中可能形成的复杂攻击路径。

7.7.1 流程

漏洞链分析侧重于识别和分析多个安全漏洞如何被串联利用来达成攻击目的。通过理解漏洞如何相互作用，可以更有效地评估整体风险并制定防御策略。漏洞链分析帮助确定哪些漏洞组合构成最高风险，并应优先修复。

第一阶段-识别和分类：

使用自动化工具和手动测试方法以及供应链的SBOM中识别系统存在的各类漏洞，包括软件缺陷、配置错误、不安全的编程实践等。根据漏洞的性质和潜在影响，对漏洞进行分类，例如信息泄露、权限提升、拒绝服务等。

第二阶段-分析漏洞间的关联和作用：

分析不同漏洞之间的依赖关系，确定哪些漏洞是其他漏洞利用的前提条件。研究漏洞如何相互作用，如一个漏洞可能使攻击者获得初步的系统访问权限，而另一个漏洞则允许进一步的系统控制。

评估漏洞链中各个漏洞的联合影响，确定其对系统整体安全性的威胁程度，分析攻击者利用漏洞链进行攻击所需的技能和资源，评估攻击的可行性和潜在影响。

第三阶段-构建漏洞利用链：

确定漏洞利用的可能路径，即攻击者可能采取的步骤序列来利用这些漏洞。使用图形或模型来表示漏洞链，明确显示每个漏洞在攻击路径中的位置和作用。

第四阶段-漏洞链分析报告

详细记录漏洞链识别和评估的过程、发现的漏洞组合及其风险，提供相应的防护建议。

保存所有针对漏洞链的修复和防护措施的实施过程，确保系统安全性的持续提升。

7.8 网络安全功能测试

网络安全功能测试是验证嵌入式软件中各项安全功能是否按照设计要求正确实现和有效运行的过程，确保系统具备预期的防护能力。

7.8.1 功能性测试

- 授权：验证最小权限原则、角色/权限划分、敏感操作权限管控等是否合理有效。
- 数据加密：检查加密算法合规性、密钥管理机制（密钥生成、存储、销毁）及传输加密等是否符合安全要求。
- 日志记录和审计：验证日志的完整性、敏感信息的保护（如脱敏处理）、以及对关键操作的可追溯性等。

7.8.2 边界条件测试

- 异常输入及恶意攻击场景：对安全功能输入超长、非法或畸形数据，模拟DDOS、SQL注入、缓冲区溢出等恶意攻击，测试系统在极端情况下的安全防护能力。
- 系统异常和故障恢复：在断电、网络断连、系统崩溃或重启等情况下，验证安全功能（例如认证、日志、加密等）是否能够正确恢复和保持预期的安全性。
- 资源极限状态：如 CPU/内存耗尽时，系统的安全功能（尤其是身份验证、访问控制、加密传输等）是否仍能维持或有安全降级策略（如只接受管理接口，不接受普通用户请求）。

7.8.3 数据交互测试

- 数据完整性：验证在设备间或模块间传输过程中，是否通过校验、加密、签名等手段保证数据不被篡改；若篡改或丢包，系统是否能够及时检测或处理。
- 加密传输与防窃听：检查网络传输通道（如TLS/SSL、VPN等）的加密强度，确保数据在传输过程中无法被窃听或伪造。
- 跨组件/跨系统安全策略：若系统依赖第三方服务或组件，需验证在跨系统的数据交互中权限、密钥、会话管理等安全策略的一致性，防止由于集成不当引发漏洞。

7.8.4 安全性强化测试（可选/推荐）

- 配置安全性检查：核查系统配置文件、访问权限、日志和审计策略等，确保其符合安全基线要求。

8 医疗器械更新

8.1 风险可追溯

风险可追溯性是指在整个软件生命周期中，对漏洞风险的识别、评估、处理和监控过程进行全面记录和跟踪，确保每一个风险都能被有效管理和控制。

8.2 更新控制

更新控制是指对已识别漏洞的修复和缓解措施进行有效的管理和实施，确保软件系统能够及时应对新出现的安全威胁和漏洞。

对于已识别的漏洞，应制定具体的修复和缓解措施，包括软件更新、配置更改、硬件更换或增加安全防护措施等。建议的修复策略应详尽说明操作步骤、预期效果及可能的副作用，同时应定期复审和更新这些策略，以适应新出现的威胁和漏洞。

在涉及现成软件（OTS软件）时，更新控制尤为重要。制造商应密切关注OTS软件供应商发布的安全公告和更新补丁，确保及时获取并评估这些更新对医疗器械的影响。在实施OTS软件更新之前，应进行充分的风险评估和验证测试，确认更新不会对设备的性能和安全性产生负面影响。

当OTS供应商发布安全更新补丁后，若制造商无法及时更新软件系统，建议制造商公布风险安全公告并提供临时安全解决措施，确保OTS漏洞不会影响到患者的安全和数据的安全。

通过风险的可追溯性和更新控制，医疗器械制造商能够确保嵌入式软件系统在整个生命周期内保持高水平的安全性，及时应对新出现的安全威胁和漏洞，保障患者的安全和数据的完整性。

9 医疗器械维护

在医疗器械的全生命周期内，为确保网络安全与功能的持续有效，医疗器械注册申请人应对系统中已知和新发现的漏洞进行持续识别、评估、修复和跟踪。 并结合风险管理综合分析剩余漏洞对产品安全性方面的影响，确定补偿剩余漏洞的网络安全策略，制定剩余漏洞维护方案。

9.1 维护策略

维护策略是指在软件系统运行过程中，对已识别与新发现的漏洞进行持续管理与处置的整体计划与方法。该策略不仅要考虑漏洞本身的风险等级和影响范围，还需结合医疗器械在临床与业务层面的重要性，综合制定风险优先级和资源分配方案。维护策略通常包括以下几个要素：

9.1.1 风险评估

- **定期进行脆弱性评估：**通过漏洞扫描、渗透测试、SBOM（软件物料清单）更新等方式，识别新出现的漏洞和潜在安全风险。
- **综合分析：**从漏洞的危害程度、利用难度、影响范围以及设备功能关键性等维度，评估其对系统整体安全和临床使用的影响。

9.1.2 风险处理计划

- **修补措施：**针对已识别的漏洞制定修补方案，包括软件补丁、配置加固、硬件升级等。
- **缓解策略：**对于暂时无法彻底修复的漏洞，应制定临时或长期的安全缓解方案，如访问限制、网络隔离、监控警报等。
- **资源分配：**明确信息安全团队与研发团队的分工和职责，合理调配人力、时间与预算。

9.1.2 风险处理计划

- **明确风险接受条件：**如果短期内无法修复高危漏洞，需评估其带来的安全风险是否在可接受范围内，同时建立定期复审机制，避免长期忽视高危风险。
- **合规性考量：**确保风险接受策略符合法规和标准要求，不因暂缓修补而违反监管规定或影响临床安全。

9.2 已知漏洞维护过程

基于PDCA循环的理念，已知漏洞的维护可分为“计划（Plan）—实施（Do）—检查（Check）—改进（Act）”四个主要阶段，以确保漏洞能够得到及时、有效的处理和管理：

过程	描述
计划	● 风险评估(脆弱性评估) ● 制定风险处理计划(修补程序) ● 风险接受(什么构成接受?高危漏洞已解决/符合法规要求?)
实施	实施风险处理计划(修补/缓解策略)
检查	持续监控/审查风险
改进	维护和改进流程(在发现新风险时更新扫描配置文件和缓解策略)

9.2.1 计划

1) 风险评估（脆弱性评估）：

定期使用漏洞评估工具进行脆弱性扫描，识别系统中的新漏洞和风险。

结合业务需求和系统关键性，评估每个漏洞的风险等级和潜在影响。

2) 制定风险处理计划（修补程序）：

根据风险评估结果，制定详细的修补计划，明确修复优先级和具体措施。

确定修补措施的时间表和资源分配，确保高风险漏洞得到优先处理。

3) 风险接受：

对于无法立即修复的高危漏洞，评估其对系统安全性的影响是否在可接受范围内。

制定风险接受标准，明确何种条件下可以接受特定漏洞的存在，确保符合相关法规和标准要求。

9.2.2 实施

1) 实施风险处理计划（修补/缓解策略）：

按照风险处理计划，实施漏洞修补措施，包括软件更新、配置更改、硬件更换等。

确保修补过程的规范化和可追溯性，记录每一个修补步骤和实施细节。

9.2.3 检查

1) 持续监控/审查风险：

建立持续的风险监控机制，实时监测系统的安全状态和漏洞修复情况。

定期审查已修复漏洞的有效性，确保修补措施达到预期效果。

9.2.4 改进

1) 维护和改进流程：

根据新发现的风险和漏洞，及时更新扫描配置文件和缓解策略，确保系统能够应对最新的安全威胁。

通过持续的流程改进，提高漏洞管理的效率和效果，推动整体系统安全性的提升。

10 风险评估

风险评估是医疗器械嵌入式软件漏洞评估方法中的核心环节，旨在识别、分析和管理系统中的安全风险，确保整体系统的安全性和可靠性。鉴于医疗器械对人身安全的重大影响，风险评估必须将对患者安全的影响纳入考量，并结合国际标准ISO 14971、IEC 62304和ISO/IEC 27001。

1.风险识别：通过系统分析、威胁建模方法和漏洞识别，确定系统中的关键资产，包括敏感数据、硬件组件，以及直接影响患者安全的功能模块。识别所有可能的风险源，特别关注可能对人身安全造成影响的漏洞和威胁。

2.风险分析：利用定性和定量分析方法，如CVSS评分和故障树分析，对识别的风险进行分类、优先级排序和数值化评估，确定风险的严重性和发生概率。评估以下维度：

- 严重程度（Severity）：评估风险对患者安全和健康的潜在影响程度，包括可能导致的伤害或危害等级。根据GB/T 42062-2022，对严重程度进行分类，如轻微伤害、可逆伤害、不可逆伤害或死亡。
- 影响范围（Impact Scope）：确定风险可能影响的范围，包括单个患者、多名患者、医护人员或整个医疗系统。评估风险对数据完整性、系统可用性和业务连续性的影响。
- 利用难度（Exploitation Difficulty）：评估漏洞被利用的难易程度，包括技术复杂性、所需资源、攻击者技能水平和现有防护措施的有效性。

采用定性和定量分析方法，如CVSS评分、故障树分析和风险矩阵，将上述维度综合评估，确定风险的优先级和紧迫性。

在确定漏洞风险等级时，应综合考虑以下要素：

- 漏洞风险等级：依据CVSS评分，结合CWE分类，评估漏洞的严重性。
- 利用难易程度：评估攻击者利用漏洞所需的技术能力和资源。

- 潜在影响范围：分析漏洞被利用后对系统、数据及患者安全的可能影响。
- 系统关键性：根据资产和资产组的业务关键性评级，评估漏洞对系统关键功能的影响。

根据风险评估结果，按照业务风险进行优先排序。高风险漏洞应优先处理，以降低系统整体安全风险。具体步骤包括：

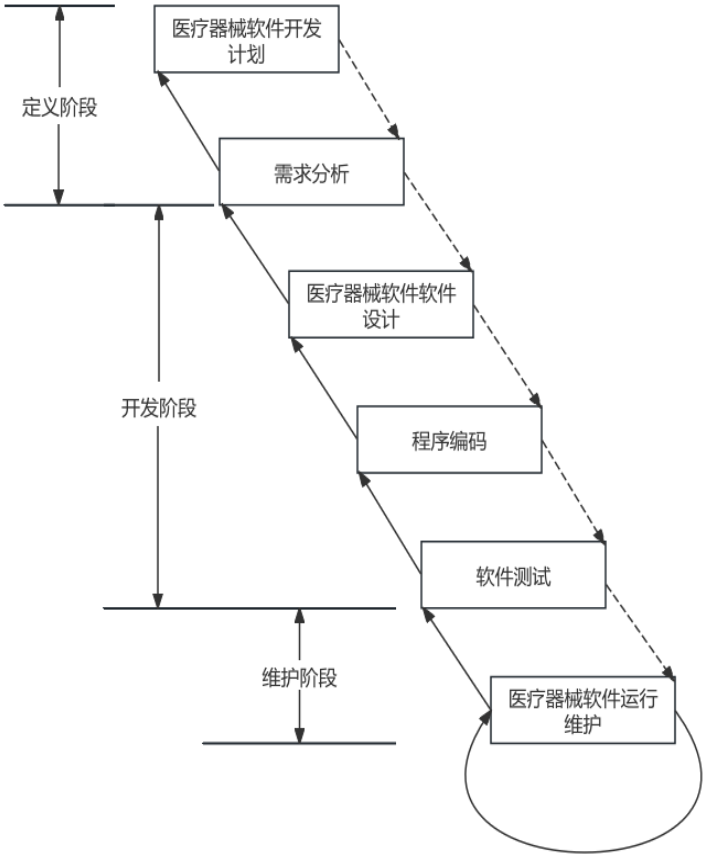
- 漏洞识别：通过漏洞评估工具计算资产的业务风险。
- 优先级分配：根据风险等级和业务关键性，分配修复优先级。
- 资源分配：根据优先级分配修复资源，确保高风险漏洞得到及时处理。

3.风险控制：通过技术和管理措施，如加密、访问控制、安全培训以及应急响应计划，降低风险的发生概率或影响程度。风险控制策略包括风险的消除、降低、转移和接受。

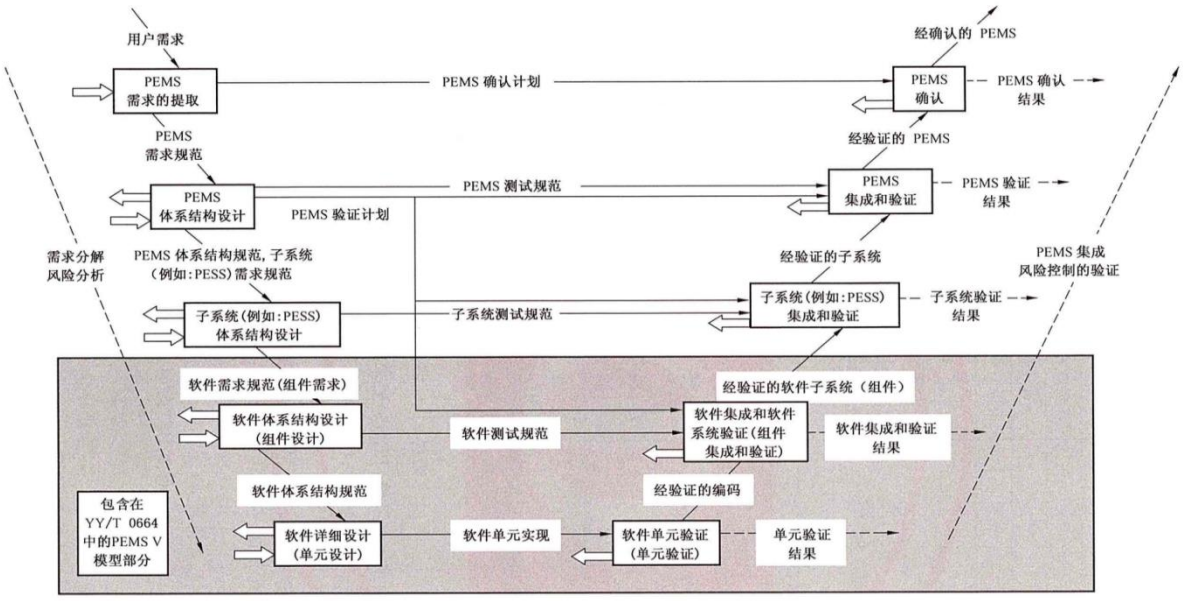
4.风险接受：在风险可接受范围内，决定不采取进一步的控制措施，并记录风险接受的依据和条件。确保所有被接受的风险都在可控范围内，并具备相应的监控和应对措施。

5.风险评估报告：编制风险评估报告，记录风险识别、分析、控制和接受的结果，提供系统的安全管理和决策依据。报告通过内部和外部审查，确保其有效性和合规性。

附录 A 瀑布模型（Waterfall Model）示意图



附录 B V模型（V-Model）示意图



附录 C 敏感数据典型示例

类别	典型示例
生物识别信息	个人基因、指纹、声纹、掌纹、眼纹、耳廓、虹膜、面部识别特征、步态等
宗教信仰信息	信仰的宗教、加入的宗教组织、宗教组织中的职位、参加的宗教活动、特殊宗教习俗等
特定身份信息	犯罪人员身份信息、残障人士身份信息、特定工作信息（如军人、警察）、身份证件号码等
医疗健康信息	病症、住院志、医嘱单、检验报告、检查报告、手术及麻醉记录、护理记录、用药记录、生育信息、家族病史、传染病史等
金融账户信息	银行、证券、基金、保险、公积金等账户的账号及口令，公积金联名账号、支付账号、银行卡磁道数据（或芯片等效信息）以及基于账户信息产生的支付标记信息等
行踪轨迹信息	实时精准定位信息、GPS 车辆轨迹信息、交通行程信息、特定住宿信息等
不满十四周岁未成年人个人信息	不满十四周岁未成年人的个人信息
身份鉴别信息	登录口令、支付口令、账户查询口令、交易口令、动态口令、口令保护答案等
其他敏感个人信息	网页浏览信息、婚史、性取向、通信内容、民族、宗教信仰、征信信息、未公开的违法犯罪记录等

参考文献

[1] GB/T 30276-2020 信息安全技术网络安全漏洞管理规范[S]

[2] GB/T 28458-2020 信息安全技术—网络安全漏洞标识与描述规范[S]

[3] GB/T 30279-2020 信息安全技术网络安全漏洞分类分级指南[S]

[4] 信息安全技术 敏感个人信息处理安全要求（征求意见稿）[S]

[5] 医疗器械网络安全注册审查指导原则[S]

[6] ISO/IEC 29147:2018, Information Technology-Security Techniques - Vulnerability Disclosure[S]

[7] ISO/IEC 30111:2013, Information Technology - Security Techniques - Vulnerability Handling Processes [S]

[8] Building a Vulnerability Management Program - A project management approach [S]

[9] FDA Guidance-Device-Cybersecurity-Premarket [S]

[10] IEC 81001-5-1 Health software and health IT systems safety,effectiveness and security- Part5-1:Security-Activities in the product life cycle [S]

[11] Framing Software Component Transparency: Establishing a Common Software Bill of Material (SBOM) [S]

[12] PLAYBOOK FOR THREAT MODELING MEDICAL DEVICES [S]

[13] FDA Off-The-Shelf Software Use in Medical Devices [S]

[14] GIAC Building a Vulnerability Management Program -A project management approach