Computer Fraud and Security
ISSN (online): 1873-7056

Cybersecurity-Centric Medical Device Firmware Development: The Role of IEC 62304 and Embedded Real-Time Operating Systems (RTOS) in Ensuring Safety and Compliance

Shiva Kumar Madishetty

Mentor, OH, USA shiva.madishetty1@gmail.com

Shilpa Chakinala

Raleigh, NC, USA chakinala.shilpa@gmail.com

***Abstract*—**This paper presents a comprehensive exploration of cycbersecurity aspects for medical devices developed under IEC 62304 and RTOS frameworks.

When building software for medical devices, security matters a lot. Following IEC 62304 means planning every stage carefully. Because mistakes can harm patients, each update gets reviewed thoroughly. Instead of treating safety and function separately, they work together from start to finish. With threats always changing, checks happen continuously. Even small code changes follow strict rules. So risks drop when design meets real world demands early. Trustworthiness comes first, then meeting legal standards. The process opens with spotting key risks early

One big worry for tiny computers inside medical gear is when bad code sneaks in. Hacking through repeated signals happens too, messing up normal operation. Access without permission opens more doors than it should. These risks change how devices behave in quiet but serious ways. Regular upgrades to device software matter because actual breaches show what happens without strong defenses. This report examines events where weak protection led to serious problems. One example follows another showing how fast threats spread when systems lack updates. Each case reminds us that small gaps invite big risks. Attention to detail grows more critical after seeing repeated failures. What seems minor often becomes major through neglect. Real damage occurs even when warnings were clear beforehand.

Building things comes first. Then keeping them running matters just as much. Staying alert for what could go wrong shapes how work moves forward. That part matters inside worldwide rules like those from the FDA or the EU MDR and ISO standards. The paper evaluates the limitations of bare, metal systems. Yet shows what RTOS setups do when problems appear by locking down timing, keeping errors separate, starting up safely, yet connecting each piece tightly.

One step back reveals how systems hide complex details. FreeRTOS stands out when looking at approval ratings across industries. Zephyr follows close, tied strongly to safety checks built for hospitals. Security steps up differently under QNX, known more for steady performance than flash. Then there is INTEGRITY, fitting niches where rules are strictest. Each year shifts what matters most, 2025 reshapes priorities again

Looking at FDA cybersecurity advice shows what it means for, One part involves firmware setup, along with checking possible risks. Another piece tracks what components are used, making sure records stay up to date. Keeping updates safe while managing risks in how products are delivered.

Next up, ways to weave cybersecurity into RTOS work that follows IEC 62304 are laid out in the report. Writing code that stays safe. Making real, time operating systems tougher against threats. Fixing known issues regularly through updates. Yet checking methods include things like static or dynamic review, plus trying out security through penetration tests, along with strict proof using formal verification. Examples drawn from real situations show how these work

Folks like Ottobock or Innolitics, then again, actual events show similar patterns. Moments from hospitals, moments from labs, each pointing in one direction without saying it outright. Buzzing machines in hospital rooms can fail when signals twist. Hidden flaws appear through quiet shifts in beeping patterns. Devices meant to protect sometimes open backdoors instead. Weak spots emerge not from design but how pieces

connect

Finding new paths matters most. Looking ahead shapes what comes next. Some ideas point toward tomorrow. What follows builds on earlier steps. Thoughts turn to what might be. Later sections examine where things could go.

Finding threats using smart tools runs alongside cloud and local systems working together. Security logs live on a shared chain so changes show up fast. Trust nothing by default shapes how access gets managed every step of the way. Virtual copies mirror real devices while testing odd behaviors ahead of time. New math guards data when future computers grow too powerful. Secret codes help lock down device software, making tough protection a key part of next, generation surgical and diagnostic systems.

I. Introduction

The proliferation of connected medical devices has made them integral to modern healthcare, playing a critical role in diagnosing, monitoring, and treating patients. As these devices increasingly rely on embedded software and network connec- tivity, they become susceptible to cybersecurity threats that can compromise patient safety and data integrity. Traditional med- ical device development focused heavily on functional safety, often underestimating the implications of cyber risk. However, a series of high-profile security breaches and a corresponding increase in regulatory oversight have elevated cybersecurity to a primary concern in medical software engineering.[10]

IEC 62304, the international standard for medical device software life cycle processes, provides a structured frame- work for developing safe and reliable software. It establishes classifications based on potential hazard levels and prescribes activities for software development, maintenance, and risk management. The integration of secure embedded Real-Time Operating Systems (RTOS) into IEC 62304-compliant work- flows offers a robust defense against software-based attacks and system failures. This paper examines the intersection of RTOS design, cybersecurity best practices, and IEC 62304 compliance, and their collective role in ensuring the reliability of medical software.

II. Cybersecurity Threats in Embedded Medical Devices

Embedded systems in medical devices are increasingly network-connected and reliant on real-time software, thereby expanding their attack surface and making them attractive targets for cyber threats. These threats pose significant risks not only to data confidentiality and system availability but also to patient safety. Table I summarizes common threats and their potential impact on embedded medical systems. Key cybersecurity challenges in this domain include the following:

- **Attack Vectors:** Connected medical devices face a range of security threats, including buffer overflows, command injection, and unauthorized firmware updates. Multiple at- tack vectors, including buffer overflows, command injec- tion, and unauthorized firmware updates, threaten medical devices. Furthermore, insecure communication protocols (e.g., Bluetooth LE, Zigbee, Wi-Fi) and physical inter- faces such as USB ports can serve as critical entry points for malware or malicious reconfiguration [3].

- **Real-World Incidents:** The 2017 FDA recall of St. Jude Medical's pacemakers highlighted a critical vulnerability that permitted unauthorized firmware modification via radio signals, thereby exposing patients to risks of pacing manipulation [2]. Similarly, the WannaCry ransomware attack impacted over 200,000 systems, including imaging devices within the UK's National Health Service (NHS), leading to significant disruptions such as surgical cancel- lations and delays in patient care [4].

- **Embedded Risks:** Traditional security mechanisms used in IT (such as antivirus, intrusion detection, and TLS termination) are difficult to deploy in embedded systems due to resource constraints, lack of dynamic memory, and real-time timing guarantees. These systems often operate in isolated environments, complicating patch deployment and monitoring.

To illustrate the diversity and severity of potential threats, Table I categorizes common cybersecurity risks affecting em- bedded medical devices and highlights their impact vectors and system-level consequences.

In summary, the convergence of real-time embedded pro- cessing, connectivity, and physical interaction with patients makes medical devices uniquely susceptible to a hybrid of IT and operational technology (OT) threats. Addressing these vulnerabilities requires not only secure coding and protocol design, but also robust system-level architectural

protections enabled through real-time operating systems (RTOS) and lifecycle process standards such as IEC 62304.

III. IEC 62304 OVERVIEW

One version of IEC 62304 came out in 2006, later updated through changes added over time. A widely accepted guideline across countries, IEC 62304:2015 outlines how software used in medical devices should be managed throughout its entire existence. While it sets clear stages, each phase connects to responsibilities during development, testing, and updates, guiding teams through structure without dictating methods.

Working under IEC guidelines means following clear steps for building software, keeping it updated, handling risks carefully, while also managing system changes through orga- nized methods. Software must follow rules. This rule covers programs that run by themselves built right into medical equipment, along with the programs it runs system.

Well, built tools inside the machine start with careful coding steps. Each part works smoothly because testing happens early. Mistakes get caught fast since checks run often. Clear methods guide how updates grow over time. Problems show up less when planning comes first

Following each step closely helps lower dangers lurking in the process. Not every hazard gets caught, yet careful tracking cuts chances sharply. Tighter oversight slips through gaps others miss. Reducing exposure happens when details stay visible. Strong checks slow down problems before they grow

Putting patient safety or how well the machine works at risk is not an option.

A. Core Requirements and Processes

The standard organizes the software life cycle into five key processes:

- **Software Development Process:** This includes require- ments gathering, architectural design, detailed design, coding, integration, and unit/system verification. All out- puts must be fully documented and linked to their corre- sponding inputs for traceability.
- **Software Maintenance Process:** Defines structured methods for introducing changes, patches, and feature updates while ensuring continued safety and compliance. Impact analysis and regression testing are mandated.
- **Software Risk Management Process:** Strongly tied to ISO 14971 [31], this process requires continuous iden- tification, analysis, control, and verification of software risks, especially for Class B and Class C devices.
- **Software Configuration Management Process:** Ensures version control of code and documents, tracks changes, and maintains reproducibility for audits and certification.
- **Software Problem Resolution Process:** Involves defect tracking, root cause analysis, and corrective actions. This aligns with ISO 13485 quality management requirements.

TABLE I

COMMON CYBERSECURITY THREATS IN MEDICAL DEVICES

Threat	**Attack Vector**	**Potential Impact**
Code Injection	Network interfaces, firmware entry points, serial ports	Execution of unauthorized code, device control hijacking, malfunction in critical operations
Replay Attacks	Wireless communication (BLE, Wi-Fi), un- secured protocols	Injection of outdated or manipulated data, misleading sensor readings, false alarms
Denial-of-Service (DoS)	TCP/IP stack, malformed inputs, network flooding	System crashes, therapy interruption, increased patient risk, reboot cycles

Unauthorized Updates	OTA updates, USB ports, misconfigured bootloaders	Installation of rogue firmware, bricked devices, permanent loss of functionality
Privilege Escalation	Operating system bugs, insecure APIs, weak authentication controls	Unauthorized access to safety-critical functions or patient records

B. Software Safety Classification

Devices are classified based on the severity of harm that could result from a software failure:

- **Class A:** No injury or damage to health is possible.
- **Class B:** Non-serious injury is possible.
- **Class C:** Serious injury or death is possible.

Higher classification levels (B and C) demand stricter documentation, more rigorous testing, and formal verification procedures.

C. Role in Regulatory Compliance

IEC 62304 plays a central role in meeting global regulatory requirements:

- In the United States, the FDA recognizes IEC 62304 as a consensus standard under its 510(k) and PMA pathways for software as a medical device (SaMD) and embedded firmware [32].
- In Europe, compliance with IEC 62304 is essential for CE marking under the Medical Device Regulation (EU) 2017/745. It supports conformity assessment routes and is often reviewed by Notified Bodies.
- In Canada and Japan, national health regulators have adopted IEC 62304 as part of their harmonized approach to software risk management and lifecycle control.

In all cases, demonstrating conformity to IEC 62304 strengthens the overall submission package by offering proof of process maturity and safety diligence.

IV. Need for RTOS in Embedded Systems

As medical devices grow more complex and well intercon- nected, the necessity for a robust and deterministic software execution environment becomes paramount. Real-Time Oper- ating Systems (RTOS) are critical in embedded systems that must meet strict timing, safety, and reliability constraints. Un- like the general-purpose operating systems, RTOS platforms are optimized for predictability and fine-grained control over task execution, resource allocation, and fault tolerance.

Few people notice how machines inside hospitals keep changing shape. Devices talk to each other now, unlike before. Complexity sneaks in without warning. Connections multiply through wires and signals. Each upgrade adds unseen layers. Technology shifts while attention wanders elsewhere the need for a robust and deterministic software execution.

When things move fast, timing matters most. These systems rely on split, second precision. RTOS steps in where delays cause problems. Embedded tech often depends on instant responses. Missing a beat can change outcomes. Speed shapes how devices behave. Every millisecond counts when reactions need to be immediate.

Timing matters most, then comes safety, followed by how well it holds up. General systems juggle many tasks without such tight rules; these specialized ones focus on speed when it counts. Steady outcomes come with precise handling of each step. One thing follows another without surprise, yet every move stays sharp under close watch how resources get shared, yet faults stay managed, mainly because of their specialized kernel designs.

A. Challenges in Bare-Metal Systems

Traditional bare-metal systems (i.e., software running with- out an OS) are adequate for simple applications with limited concurrency. However, they present several limitations in modern embedded systems:

- Lack of task isolation increases the risk of cross- component failures.
- Scalability is limited due to tightly coupled code and shared control logic.
- Interrupt-driven designs grow unmanageable as the num- ber of features increases.
- Integration of connectivity (Wi-Fi, BLE), user interfaces, and security becomes highly complex.

Collectively, these limitations significantly impede the de- velopment of safe, maintainable, and secure medical devices, necessitating a more robust software foundation.

B. RTOS as a Solution

RTOS platforms solve these challenges by providing:

- **Deterministic Task Scheduling:** Guarantees real-time behavior with preemptive, round-robin, or fixed-priority schedulers to meet timing deadlines.
- **Multitasking Support:** Enables simultaneous execution of user interface, sensor acquisition, communication pro- tocols, and safety-critical logic with task prioritization.
- **Hardware Abstraction:** Simplifies driver management, peripheral interaction, and CPU porting.
- **Fault Isolation:** Through memory protection and sepa- ration kernels, RTOSes help contain faults and prevent their propagation, thereby enhancing system stability and security by limiting the impact of compromised compo- nents.
- **Security Integration:** Features like secure boot, sand- boxing, and cryptographic services are built into certified RTOS platforms for regulated domains.
- **Regulatory Support:** RTOS platforms often come with documentation, traceability, and test evidence to support IEC 62304, ISO 26262, DO-178C, and FDA submission requirements.

C. Application Examples in Healthcare

RTOS-enabled embedded systems are used in a variety of medical applications:

- Infusion pumps with alarm management and real-time drug delivery
- Cardiac monitors with ECG signal pro

V. Role of Embedded RTOS in Secure Medical Devices

Real time operating systems underpin device operations crucial when shaping embedded medical tools, because it guarantees safety standards hold up under real world stress. Fault tolerant operations run on precise schedules. Timing stays locked, no matter what unfolds. Systems hold firm when stress appears. A chunk of memory gets set aside first. Not like regular OS setups, where everything mixes together. Timing precision shapes how RTOS platforms operate, while predictability stands central to their function high assurance, characteristics that are essential for patient therapy works well when rules are followed. Safety shows up where care meets standards. Rules stick around because they keep things on track.

Devices like ventilators operate alongside infusion pumps. Wearable ECG monitors function together with robotic surgery tools. These machines depend on steady performance. Robotic systems assist during complex procedures. Infusion pumps deliver medication precisely. Ventilators support breathing when needed. Each device runs using specialized software. Monitoring tools track heart activity continuously. Surgical robots move with high accuracy. All require regular mainte- nance. Performance issues can disrupt care. Reliability matters most in critical moments

RTOS to manage concurrent software tasks while ensuring resilience against failure comes hand in hand with strong defenses online. When one part breaks, others keep running, security stays tight even under pressure.

A. Key Security and Safety Features of RTOS

Security gets stronger when real, time systems weave to- gether key protections. These tools work behind the scenes, keeping devices stable. Instead of just reacting, they prevent issues before harm occurs. A mix of checks guards against

tampering. Each layer adds resilience without slowing perfor- mance down.

- **Memory Protection Units (MPUs):** FreeRTOS and Zephyr use MPUs for memory protection. One task stays separate from another. Because of that, errors cannot spread. Unauthorized memory access is part of it, a key method used to reduce risks during execution, such as when buffers overflow.

- **Secure Boot Mechanisms:** Platforms such as IN- TEGRITY and QNX avail hardware-assisted secure boot, verifying cryptographic signatures of bootloaders and kernel images. This mechanism promises that only trusted software is executed, thwarting persistent malware and firmware tampering. QNX uses hardware to help confirm bootloaders and the kernel are signed correctly through cryptography. Photos help confirm just approved pro- grams run, blocking stubborn viruses plus changes deep in the system [48].

- **Deterministic Scheduling:** Medical RTOS kernels use preemptive or fixed-priority scheduling to guarantee bounded task execution times. This is vital for Class C systems where task latency can directly affect patient health (e.g., real-time insulin dosing or defibrillator signal analysis).

- **Multilevel Security (MILS):** Green Hills INTEGRITY RTOS employs a partitioned architecture with strict separation of kernels to isolate critical and non-critical components. This MILS design enables certified medical systems to run diagnostics, network stacks, and GUIs without compromising the therapy core [50].

- **Certifiability and Regulatory Alignment:** Many RTOSes offer certification-ready packages with design artifacts, traceability matrices, and test coverage reports. For instance, SAFERTOS offers pre-certified modules for IEC 62304 compliance, and QNX has achieved IEC 62304, ISO 26262, and POSIX certification

Table II compares RTOS options commonly used in safety- critical medical devices.

VI. FDA 2025 Cybersecurity Guidance: Implications for Firmware and Embedded RTOS in Medical Devices

The 2025 FDA guidance titled *"Cybersecurity in Med- ical Devices: Quality System Considerations and Content of Premarket Submissions"* [38] provides a comprehensive framework for securing medical devices throughout the Total Product Lifecycle (TPLC). A key emphasis is placed on secure firmware development practices, especially in devices that rely on embedded real-time operating systems (RTOS). As connectivity becomes a defining characteristic of modern med- ical technologies, the intersection of firmware security, RTOS design, and regulatory compliance is increasingly critical.

A. Cybersecurity as a Core Safety Consideration

Besides physical risks, digital threats matter just as much. Safety includes protection from hacking. The agency sees

TABLE II

RTOS Comparison for Medical Device Applications

RTOS	**MPU**	**Certifications**	**Secure Boot**	**Typical Usage**	**Footprint**	**Scheduling**	**Commercial Support**
FreeRTOS	Yes	IEC 62304 [5]	Add-on	Wearables, Pumps, Monitors	<20 KB	Preemptive	Yes [6]
Zephyr	Yes	Partial IEC 62304 [7]	Yes	IoT health sensors, BLE medical hubs	∼50–80 KB	Preemptive	Yes (LF, Nordic) [8]
QNX	Yes	ISO 26262,	Built-in	Imaging, ventilators,	>100 KB	Microkernel	Yes

		IEC 62304 [9]		di- agnostics			(QNX) [10]
INTEGRIT Y	Yes	DO-178C, IEC 62304 [11]	Built-in	Robotic surgery, life-support systems	>150 KB	Fixed-priority	Yes (Green Hills) [12]

cyber dangers as part of overall harm prevention. Protection isnt optional anymore. Functionality ties directly to secure design. A flaw in code can be as dangerous as a broken wire. Devices must resist tampering by default. Risk grows when systems connect online. Hidden weaknesses might cause real injuries. Security shapes how well tools perform. The FDA treats weak software like any other defect.

After market release, upkeep becomes an issue. As required by the rules, that manufacturers embed cybersecurity into the design control process per 21 CFR Part 820, integrating threat modeling. Putting risk checks alongside ways to reduce problems straight into the work flow development lifecycle [38].

B. Secure Product Development Framework (SPDF) and IEC 62304 Alignment

To meet these expectations, manufacturers are encouraged to implement a Secure Product Development Framework (SPDF), which parallels the processes defined in IEC 62304. This includes:

- Incorporating security risk management as part of design controls (21 CFR 820.30).
- Performing dual-layered risk assessments, encompassing both safety (per ISO 14971) and cybersecurity consider- ations.
- Establishing design inputs, outputs, and acceptance cri- teria for security features such as secure boot, encrypted communication, and role-based access controls.
- Documenting all cybersecurity-relevant design decisions in the Design History File (DHF) and Device Master Record (DMR).

C. Firmware-Level Requirements and RTOS Design Consid- erations

The guidance articulates multiple firmware-specific respon- sibilities that directly impact how RTOS-based devices are developed and maintained:

1) **Threat Modeling:** Must be considered throughout the architecture process and include any assumptions about hostile environments, such as hospital networks or com- promised supply chains. Models should identify vul- nerabilities across all firmware layers, including RTOS kernels, APIs, and middleware.

2) **Security Architecture Views:** FDA recommends in- clusion of detailed architectural diagrams that delineate trust boundaries, memory partitions, and control/data flow between firmware modules and connected systems [38].

3) **Code, Data, and Execution Integrity:** RTOS firmware must enforce execution integrity through techniques such as signed firmware images, memory protection units (MPUs), watchdog timers, and secure bootloaders, which help ensure that only authenticated code runs, prevent memory corruption, detect system hangs, and protect the boot process from tampering. Design outputs must verify these controls function as intended and include validation evidence in premarket submissions.

4) **Software Bill of Materials (SBOM):** The guidance mandates a comprehensive SBOM, including all third- party libraries, drivers, and open-source components in- tegrated into the RTOS firmware. It must document the support status and known vulnerabilities of each compo- nent, especially those that fall under FDA-defined “cyber devices” as per Section 524B of the FD&C Act [38].

5) **Patchability and Updates:** Devices must include capa- bilities for secure, cryptographically verifiable firmware updates (e.g., OTA or USB). The update mechanism should support rollback protection and logging for audit purposes. FDA also recommends metrics such as mean- time-to-patch and patch uptake rate be tracked and sub- mitted for

review.

6) **Testing and Anomaly Assessment:** Cybersecurity test- ing is not limited to functional validation. FDA re- quires fuzz testing, penetration testing, and static/dynamic code analysis at the firmware level. Unresolved anoma- lies—particularly those involving undefined behavior or undocumented exceptions—must be evaluated for se- curity implications and included in the premarket risk assessment documentation [38].

D. Supply Chain and Interoperability Risks in RTOS Firmware

The 2025 guidance stresses that device manufacturers are re- sponsible for ensuring the integrity and security of third-party software, including RTOS packages and hardware abstraction layers (HALs). It highlights that:

- Vendors supplying RTOS platforms must adhere to robust quality systems and demonstrate update mechanisms for vulnerabilities.
- Interoperability with hospital networks, EHRs, or mobile devices must be evaluated for risks such as man-in-the- middle (MITM) attacks, protocol downgrades, or insecure credential storage.
- Firmware design must incorporate runtime monitoring hooks and event logging mechanisms that are protected against tampering.

E. Documentation and Submission Expectations

Premarket submissions (510(k), PMA, De Novo) must now include:

- Full security architecture documentation, including dia- grams of firmware subsystems.
- Security risk management reports aligned with AAMI TIR57 and ANSI/AAMI SW96.
- Threat models, attack trees, and traceability matrices linking firmware functions to mitigations.
- Cybersecurity labeling for users and IT departments, in- cluding device hardening instructions and update policies.

F. Metrics for Compliance Monitoring

To ensure continuous improvement, FDA recommends man- ufacturers track metrics such as:

- Percentage of firmware vulnerabilities patched over time.
- Time from vulnerability disclosure to patch deployment.
- Field uptake rate of critical firmware updates across device installations.

These metrics are intended to be submitted in premarket dossiers and PMA annual reports, providing regulators with insight into the effectiveness of the cybersecurity program.

G. Case Study: IEC 62304 in Practice

A leading European manufacturer of infusion pumps imple- mented IEC 62304 as part of a product redesign after a Class

II FDA recall due to firmware instability. By reclassifying the software as Class C, the team adopted stricter verification methods, including unit testing, hazard-based traceability, and integration with static analysis tools like Polyspace. After a full lifecycle reset and recertification effort, the company received CE Mark approval and achieved smoother entry into Asian markets that recognized IEC 62304.

This case underscores the standard's role not just in risk mitigation but also in accelerating international market access.

VII. Integrating Cybersecurity into IEC 62304-Compliant RTOS Development

The convergence of cybersecurity principles with IEC 62304-compliant real-time operating system (RTOS) develop- ment ensures that medical device software is both functionally safe and resilient against malicious threats. IEC 62304 focuses on structured lifecycle management, while secure RTOS prac- tices embed runtime protections. Together, they form a secure- by-design paradigm for medical software systems.

- **Threat Modeling:** As an initial security activity, threat modeling helps identify potential attack surfaces early in the software development life cycle (SDLC). Frameworks

such as STRIDE (Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, Elevation of Privilege) and Data Flow Diagrams (DFDs) are used to evaluate vulnerabilities in data paths, inter-process communication, and hardware interfaces. This aligns with IEC 62304's risk management processes and supports traceability of mitigations.

- **Secure SDLC Integration:** Cybersecurity processes must be embedded within each IEC 62304 phase. During software development (Clause 5.3), secure coding stan- dards (e.g., MISRA C, CERT C) are enforced. Static code analysis tools like Coverity or SonarQube, and dynamic testing via fuzzing (e.g., with AFL or libFuzzer), uncover memory corruption and logic flaws. Secure code review gates are introduced before advancing to verification. These practices reduce latent vulnerabilities in safety- critical code paths.

- **RTOS Hardening:** Hardened RTOS configurations are essential for minimizing the trusted computing base (TCB). Measures include disabling unused APIs, en- forcing memory protection via the Memory Protection Unit (MPU), and isolating safety-critical threads using privilege separation or process partitioning (e.g., via QNX microkernel or INTEGRITY RTOS). Secure inter- process communication (IPC) and encryption (e.g., TLS, AES) are recommended for communication with external devices or cloud endpoints.

- **Patch and Configuration Management:** Per IEC 62304's maintenance and configuration management clauses (5.5 and 6), patching strategies must include vulnerability triage, test validation, and secure over-the- air (OTA) mechanisms. RTOS configuration artifacts, software of unknown provenance (SOUP), and third-party libraries are versioned and tracked. Tools like SBOM (Software Bill of Materials) and secure bootloaders aid in integrity verification of software updates.

A synchronized approach where IEC 62304 process rigor complements real-time security controls allows manufacturers to meet FDA premarket cybersecurity guidance, EU MDR software classification, and post-market vulnerability handling expectations. Ultimately, it fosters the development of trust- worthy medical devices that operate safely and securely in connected environments.

Figure 1 illustrates the layered architecture that integrates cybersecurity, RTOS, and lifecycle control.

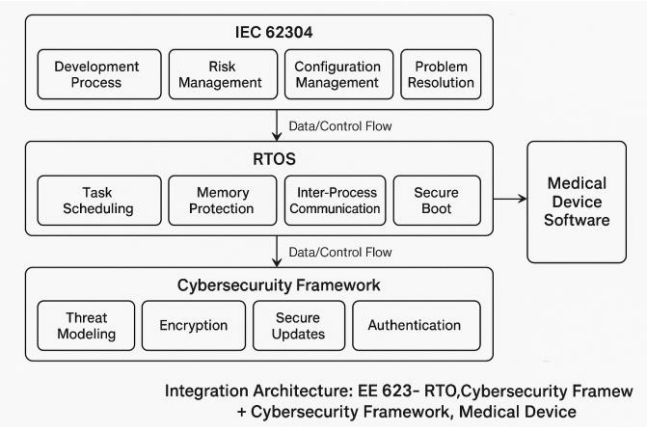

Fig. 1. Integration of RTOS, Cybersecurity, and IEC 62304 processes

VIII. Validation and Certification Strategies

Verification, validation (V&V), and certification are critical components in the development lifecycle of medical device software, especially when targeting compliance with IEC 62304, ISO 14971, and regulatory authorities such as the FDA or EU MDR [17], [31], [32]. These activities ensure that the final product meets both functional and cybersecurity requirements.

- **Static and Dynamic Code Analysis:** Static analysis tools such as Coverity, Polyspace, and Cppcheck scan the source code for potential defects, including buffer over- flows, null pointer dereferencing, and MISRA compliance violations [49]. Dynamic tools like Valgrind and Address- Sanitizer detect memory leaks and race conditions [20], [21]. These techniques align with IEC 62304 Clause 5.6 for software verification.

- **Penetration Testing and Security Validation:** Penetra- tion testing replicates attack scenarios in a controlled environment to evaluate the robustness of implemented controls. Tools like Metasploit and firmware emulation frameworks help uncover vulnerabilities in communica- tion stacks and bootloaders [22]. This is increasingly recommended in FDA premarket submissions.

- **Formal Verification (Class C Devices):** For high-risk Class C devices, formal methods such as SPIN and Frama-C can mathematically verify logic and memory safety properties, especially in safety-critical modules [23], [24].

- **Documentation:** Traceability matrices, risk control logs, versioned SBOMs (Software Bill of Materials), and se- curity testing reports are central to regulatory submission and postmarket surveillance compliance [25], [26].

IX. Case Studies in Cybersecurity-Centric Firmware Development

A. IEC 62304 Integration in a Class II Diagnostic Device

IEC 62304 provides a structured framework for software life cycle processes in medical devices. A diagnostic device manufacturer faced challenges such as lack of secure update mechanisms, missing Software Bill of Materials (SBOM), and inconsistent secure coding practices. By aligning with IEC 62304 and ISO 14971, the team implemented:

- A rigorous classification system (Class A, B, C) to assess risk levels.
- Traceability matrices linking requirements to tests and risk controls.
- Hardware-in-loop testing to simulate real-world fault con- ditions.
- Continuous integration pipelines with automated regres- sion testing.

The result was a significant reduction in patch deployment time and improved FDA feedback. The use of automated frameworks like DojoFive enabled real-time verification and documentation, ensuring compliance and enhancing patient safety [43].

B. Ottobock Prosthetics and RTOS-Based Firmware Security

Ottobock, a leader in wearable bionics, integrated a safety- critical RTOS into its prosthetic devices to meet IEC 62304 Class C requirements. Their cybersecurity strategy included:

- Deployment of SAFE RTOS® for deterministic behavior and secure boot.
- Establishment of a Security Operations Center (SOC) with 24/7 monitoring.
- Consolidation of cybersecurity tools using Microsoft De- fender and Sentinel.
- Real-time threat detection and incident response proto- cols.

This approach provided holistic transparency across Otto- bock's IT infrastructure and enabled rapid response to threats. The integration of RTOS not only improved firmware reliabil- ity but also streamlined regulatory submissions [44].

C. Innolitics Threat Modeling and FDA Compliance

Innolitics supported over 35 diagnostic devices through FDA submissions by embedding cybersecurity into the devel- opment lifecycle. Their practices included:

- STRIDE-based threat modeling to identify spoofing, tam- pering, and elevation of privilege risks.
- SBOM creation and monitoring for third-party vulnera- bilities.
- Penetration testing and fuzzing to uncover edge-case exploits.
- Lifecycle cybersecurity management plans aligned with FDA guidance.

These efforts ensured robust documentation and reduced postmarket vulnerabilities. Their approach exemplifies how threat modeling can be both practical and regulatory-compliant [45].

D. Cyberattack on ICU Heart Monitors

A real-world incident documented by HHS involved a phish- ing attack that compromised a hospital's file server connected to ICU heart monitors. The attacker rebooted all monitors, endangering multiple patients. Mitigation strategies included:

- Network segmentation to isolate critical devices.
- Asset management to track and secure endpoints.
- Restriction of internet-bound access to prevent lateral movement.
- Implementation of incident response playbooks and dig- ital signatures.

This case underscores the importance of proactive cyberse- curity planning and the need for medical device security to be integrated into hospital IT operations [46].

E. 2023 Cybersecurity Vulnerability Report

A joint study by Health-ISAC, Securin, and Finite State analyzed 966 products from 117 vendors, revealing:

- 993 vulnerabilities, with 160 weaponized and trending in the wild.
- A 437% year-over-year increase in Remote Code Execu- tion (RCE) and Privilege Escalation (PE) exploits.
- Class II devices (e.g., infusion pumps, CT scanners) had the highest vulnerability count.
- A ransomware attack led to a neonatal fatality due to disrupted fetal monitoring.

This report highlights the urgent need for embedded cy- bersecurity in firmware development and the importance of vulnerability monitoring and SBOM tracking [47].

X. Future Directions

The future of secure medical software development is being shaped by the convergence of artificial intelligence, distributed system architectures, advanced cryptographic technologies, and system-level security frameworks. As medical devices grow increasingly complex, connected, and autonomous, the need for proactive, self-defending, and standards-aligned se- curity solutions becomes more urgent. The following areas highlight emerging trends and innovations that are likely to redefine how embedded medical firmware and OR-integrated systems evolve.

- **AI for Threat Detection and Response:** Embedded artificial intelligence, particularly lightweight and energy-efficient models, will increasingly be used to monitor device behavior in real time. These models can learn baseline operational patterns and detect deviations indica- tive of cyber intrusions, firmware tampering, or hardware malfunctions. Such on-device anomaly detection systems support zero-day threat identification and can dynami- cally adjust firewall rules or network configurations in response [39]. These AI models complement traditional static defenses by introducing adaptive, behavior-based security mechanisms.

- **Edge-Cloud Hybrid Architectures:** Modern device ecosystems will leverage edge-cloud co-processing, where sensitive tasks are executed locally on secure RTOS platforms, while updates, analytics, and logging are offloaded to trusted cloud infrastructures. Secure over- the-air (OTA) updates, remote diagnostics, and teleme- try streaming must conform to authenticated, encrypted

- transport protocols—consistent with IEC TR 60601-4-5 requirements for connected medical systems [40]. This architecture not only supports lifecycle scalability across distributed devices but also enables centralized policy enforcement and fleet-wide vulnerability patching.

- **Blockchain for Secure Logging and Integrity As- surance:** Blockchain and distributed ledger technologies (DLT) provide immutable and verifiable logs that are particularly valuable for audit trails, regulatory compli- ance, and forensic investigation. By cryptographically chaining firmware update records, access control events, and security alerts, these systems ensure that critical logs cannot be tampered with—even in the event of device compromise. Potential applications include FDA- compliant update tracking, multi-party device certifica- tion trails, and transparent device provenance [41].

- **Zero Trust Architecture (ZTA) in Embedded Systems:** The zero trust paradigm—"never trust, always verify"—is increasingly being applied to embedded medical devices. This includes micro-segmentation of tasks, enforcement of least privilege at process and thread levels, and con- tinuous authentication of all inter-process and external communications. ZTA can be implemented within RTOS environments using secure enclaves, memory isolation techniques, and hardware-backed identity frameworks such as TPMs or secure elements. NIST SP 800-207 outlines a reference model for ZTA that, when adapted to embedded systems, can significantly reduce lateral movement risks and supply chain attack surfaces [42].

- **Digital Twin and Predictive Simulation:** Future OR in- tegration may involve real-time synchronization between physical surgical equipment and their virtual counter- parts—digital twins—hosted in secure, high-fidelity sim- ulation environments. These twins can be used to model failure modes, test software patches before deployment, and simulate attack scenarios, allowing for proactive mit- igation strategies and accelerated certification timelines. Integration with FDA-regulated model-based design envi- ronments could further streamline regulatory submissions and safety validation.

- **Post-Quantum Cryptography (PQC):** As quantum computing matures, conventional cryptographic methods such as RSA and ECC may become vulnerable. The med- ical device industry is beginning to explore post-quantum algorithms standardized by NIST to future-proof firmware update mechanisms, device authentication, and encrypted communications. Incorporating PQC into firmware stacks and bootloaders will be essential for maintaining long- term confidentiality and integrity guarantees, especially for devices with long operational lifespans.

These future directions are not isolated innovations but form an interdependent roadmap for transforming embedded firmware into intelligent, resilient, and verifiably secure com- ponents of the modern healthcare ecosystem. Manufacturers who anticipate and invest in these technologies early will not only meet upcoming regulatory mandates but also contribute to the broader goal of trust-driven, patient-centric medical innovation.

XI. Conclusion

The integration of advanced equipment, intelligent plat- forms, and secure software frameworks within the operating room marks a transformative era in surgical care. This pa- per has explored the convergence of interoperable surgical infrastructure, embedded real-time operating systems (RTOS), Internet of Things (IoT) devices, and artificial intelligence (AI), highlighting how each element contributes to safer, more efficient, and more adaptive patient care environments.

At the heart of this transformation lies the imperative for secure medical device software development. This re- quires a deliberate fusion of cybersecurity best practices, lifecycle standards such as IEC 62304, and robust RTOS design principles. Real-time systems must not only guaran- tee functional correctness and temporal precision but must also be engineered to withstand dynamic threat landscapes across the Total Product Lifecycle (TPLC). The 2025 FDA cybersecurity guidance further cements this paradigm shift, demanding thorough risk assessments, software bill of materi- als (SBOM), threat modeling, secure update mechanisms, and transparent lifecycle documentation. These expectations are no longer optional—they are integral to demonstrating safety and effectiveness in premarket submissions.

By embedding security into every layer of design—from surgical table automation to firmware-based secure boot and from warming cabinet telemetry to encrypted OR video feeds—developers and clinical engineers can mitigate risks ranging from system downtime to direct patient harm. More- over, the adoption of interoperable platforms enables holis- tic surgical workflows, with real-time data exchange driving decisions that reduce complications, shorten procedures, and accelerate recovery.

The successful implementation of such integrated ecosys- tems requires more than just technical alignment. It calls for a culture of collaboration between clinicians, biomedical engineers, software developers, and regulators. Each stake- holder must prioritize patient outcomes while remaining agile to evolving regulatory landscapes and cyber threats. The transition to integrated, intelligent, and secure operating rooms is not merely a technological upgrade—it is a critical step toward a safer, data-driven, and value-based healthcare future. In this context, integrated OR systems are not only opera- tionally efficient and clinically effective but also architected for resilience. The

convergence of deterministic RTOS scheduling, proactive threat modeling, interoperable hardware platforms, and lifecycle-compliant firmware development paves the way for a new standard of care—where every surgical action is supported by secure, real-time intelligence and every device interaction contributes to better patient outcomes.

REFERENCES

[1] IEC 62304:2006, ”Medical device software - Software life cycle pro- cesses,” IEC, 2006.

[2] U.S. FDA, ”Cybersecurity in Medical Devices: Quality System Consid- erations and Content of Premarket Submissions,” 2023.

[3] N. Vari et al., ”Cybersecurity Risk Management for Medical Devices,” Journal of Healthcare Informatics Research, vol. 3, no. 2, pp. 243–262, 2019.

[4] Symantec, ”WannaCry: Ransomware Attacks Healthcare,” Threat Intel- ligence Report, 2017.

[5] WITTENSTEIN High Integrity Systems, “FreeRTOS SAFERTOS Qual- ification Support,” 2022.

[6] AWS FreeRTOS Documentation. Amazon Web Services. [Online].Available: https://docs.aws.amazon.com/freertos/

[7] Zephyr Project, “Zephyr RTOS Safety Certification Project.” [Online]. Available:https://www.zephyrproject.org/zephyr-rtos-safety-certification/

[8] Zephyr RTOS. Linux Foundation. [Online]. Available: https://www. zephyrproject.org/

[9] QNX OS for Safety. BlackBerry QNX. [Online]. Available: https://blackberry.qnx.com/en/products/safety

[10] Ram Mohan Reddy Kundavaram, Rahul Reddy Bandhela, Abhishake Reddy Onteddu. (2022). AI-Driven Predictive Modeling In Healthcare: A Data Science Perspective On U.S. Healthcare Data. South Eastern European Journal of Public Health. https://doi.org/10.70135/seejph.vi.6691

[11] QNX Technology Portfolio. BlackBerry QNX. [Online]. Available: https://www.blackberry.com/us/en/products/qnx

[12] INTEGRITY RTOS Safety-Critical Certifications. Green Hills Software. [Online]. Available: https://www.ghs.com/products/safety critical.html

[13] Green Hills INTEGRITY RTOS. Green Hills Software. [Online]. Avail- able: https://www.ghs.com/products/rtos/integrity.html

[14] IEC 62304:2006, *Medical device software — Software life cycle pro- cesses*, International Electrotechnical Commission, 2006.

[15] IEC 81001-5-1:2021, *Health software and health IT systems safety, effectiveness and security — Part 5-1: Security — Activities in the product life cycle*, IEC, 2021.

[16] Amazon Web Services, ”Security and Compliance in FreeRTOS,” [On- line]. Available: https://docs.aws.amazon.com/freertos/

[17] BlackBerry QNX, ”QNX OS for Safety,” [Online]. Available: https:// blackberry.qnx.com

[18] IEC 62304:2006, “Medical Device Software – Software Life Cycle Processes,” International Electrotechnical Commission, 2006.

[19] ISO 14971:2019, “Medical Devices – Application of Risk Management to Medical Devices,” ISO, 2019.

[20] U.S. FDA, “Cybersecurity in Medical Devices: Quality System Consid- erations and Content of Premarket Submissions,” Final Guidance, Sept. 2023.

[21] MathWorks, “Polyspace Static Code Verification,” [Online]. Available: https://www.mathworks.com/products/polyspace.html

[22] Valgrind Developers, “Valgrind: A Dynamic Binary Instrumentation Framework,” [Online]. Available: https://valgrind.org

[23] G. Zaddach et al., "Avatar: A Framework to Support Dynamic Security Analysis of Embedded Systems' Firmware," NDSS, 2014.

[24] CEA List, "Frama-C," [Online]. Available: https://frama-c.com

[25] G. J. Holzmann, "The SPIN Model Checker," IEEE Trans. on Software Engineering, vol. 23, no. 5, 1997.

[26] U.S. NTIA, "Software Bill of Materials (SBOM)," [Online]. Available: https://www.ntia.gov/SBOM

[27] European Commission, "Medical Device Regulation (EU) 2017/745,"Updated 2022. [Online]. Available: https://health.ec.europa.eu/

[28] D. Chen et al., "FirmGuard: Vetting IoT Firmware via Automated Provenance Analysis," IEEE S&P, 2020.

[29] IEC TR 60601-4-5:2021, "Medical Electrical Equipment – Guidance on Cybersecurity," International Electrotechnical Commission, 2021.

[30] M. Engelhardt, "Securing Critical e-Health Data Using Blockchain Technology," Journal of Healthcare Engineering, 2017.

[31] NIST SP 800-207, "Zero Trust Architecture," National Institute of Standards and Technology, Aug. 2020.

[32] ISO 14971:2019, "Medical Devices – Application of Risk Management to Medical Devices," International Organization for Standardization, 2019.

[33] U.S. FDA, "Cybersecurity in Medical Devices: Quality System Consid- erations and Content of Premarket Submissions," Final Guidance, Sept. 2023.

[34] Medical Device Coordination Group (MDCG), "Guidance on Cyberse- curity for Medical Devices," MDCG 2019-16 rev.1, Jan. 2020.

[35] G. C. Buttazzo, "Hard Real-Time Computing Systems," Springer, 2011.

[36] J. Yiu, "The Definitive Guide to ARM Cortex-M RTOS: Building Real- Time Embedded Systems," Newnes, 2015.

[37] FreeRTOS, "Why Use an RTOS," [Online]. Available: https://www. freertos.org/about-rtos.html

[38] Green Hills Software, "RTOS for Medical Devices," [Online]. Available: https://www.ghs.com/solutions/medical.html

[39] U.S. Food and Drug Administration, "Cybersecurity in Medical Devices: Quality System Considerations and Content of Premarket Submissions," Guidance for Industry and FDA Staff, June 27, 2025. Available: https://www.fda.gov/media/172837/download

[40] J. Zhang et al., "FirmGuard: Runtime Behavior-based Firmware Integrity Checking for Embedded Devices," in *Proc. IEEE S&P*, 2023.

[41] IEC TR 60601-4-5:2017, "Medical electrical equipment — Part 4-5: Guidance and interpretation — Safety of medical electrical equipment connected to medical IT networks," International Electrotechnical Com- mission, 2017.

[42] S. Agbo, Q. Mahmoud, and J. Eklund, "Blockchain Technology in Healthcare: A Systematic Review," *Healthcare*, vol. 7, no. 2, pp. 1–15, 2019.

[43] NIST SP 800-207, "Zero Trust Architecture," National Institute of Standards and Technology, 2020. [Online]. Available: https://doi.org/10. 6028/NIST.SP.800-207

[44] DojoFive, *IEC 62304: Principles & Best Practices for Medical Device Firmware Development*, November 2024. Available: https://dojofive.com/wp-content/uploads/2024/11/ DojoFive-Whitepaper-IEC62304-ovember2024.pdf

[45] Ontinue, *Ottobock Customer Story: Cybersecurity Transformation*, July 2024. Available: https://www.ontinue.com/wp-content/uploads/2024/07/

2024 Ottobock-Customer-Story-ENG.pdf

[46] Innolitics, *Medical Device Cybersecurity: Best Practices, FAQs, and Examples*, 2025. Available: https://innolitics.com/articles/cybersecurity/

[47] HHS 405(d) Task Group, "Threat 5 – Attacks Against Connected Medical Devices," *HHS 405(d) Cybersecurity Series*, Apr. 2019. [Online]. Available: https://405d.hhs.gov/Documents/5-Threats-Series-Threat-5-Attacks-Against-Connected-Medical-Devices-Powerpoint-Updated-R. pdf

[48] Health-ISAC, Securin, Finite State, *2023 State of Cybersecurity for Medical Devices and Healthcare Systems*, 2023. Available: https://health-isac.org/wp-content/uploads/11883-StateMedSecurityReport v6.pdf

[49] QNX Software Systems and Open Compute Project Security Work- group, "Hardware-Assisted Secure Boot in Embedded Systems," *QNX OS System Security Guide and OCP Secure Boot Specifica- tion*, pp. 1–25, 2025. Available: https://www.qnx.com/developers/docs/ 8.0/com.qnx.doc.security.system/topic/manual/secure boot.html, https:// www.opencompute.org/documents/secure-boot-2-pdf

[50] OWASP Foundation, "Source Code Analysis Tools," *OWASP Commu- nity Projects*, [Online]. Available: https://owasp.org/www-community/ Source Code Analysis Tools. [Accessed: Sep. 2, 2025].

[51] Green Hills Software, *INTEGRITY Real-time Operating System*, 2025. Available: https://ghs.com/products/rtos/integrity.html